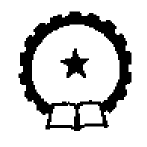
MATLAB 科学运算与工程应用丛书

# MATLAB 6.x 符号运算及其应用

刘宏友　彭　锋　等编著

博嘉科技　审



机 械 工 业 出 版 社

MATLAB 作为一种智能化高级语言，集公式演算推导与数值计算于一体，兼有出色的图形处理和数据分析功能，作为多领域的应用软件，MATLAB 的工具箱对相关学科的各种基本技术都采用了当今最先进的算法。本书共分 7 章，先简要介绍了 MATLAB 的安装及启动后，较为详细地介绍了 MATLAB 的基础知识，深入浅出地分析了 MATLAB 的符号运算功能以及绘图功能，并提供了大量的实例，便于读者自学。

本书集实用性和先进性于一体，既可作为理工科院校师生的教材和工具性参考书，也可供科研人员和工程技术人员参考。

**图书在版编目（CIP）数据**

MATLAB 6.x 符号运算及其应用/刘宏友，彭锋等编著. 一北京：机械工业出版社，2003.1

（MATLAB 科学运算与工程应用丛书）

ISBN 7-111-11572-4

Ⅰ. M… Ⅱ. 刘… Ⅲ. 计算机辅助计算一软件包，MATLAB 6.x Ⅳ. TP391.75

中国版本图书馆 CIP 数据核字（2003）第 004587 号

机械工业出版社（北京市百万庄大街 22 号　邮政编码 100037）

策　　划：胡毓坚

责任编辑：孙　业

责任印制：闫　焱

北京交通印务实业公司印刷・新华书店北京发行所发行

2003 年 2 月第 1 版・第 1 次印刷

789mm×1092mm　1/16・15.75 印张・387 千字

0001—5000 册

定价：24.00 元

# 前　言

MATLAB 原意为矩阵实验室（MATrix LABoratory），1967 年由 Clere Maler 用 FORTRAN 语言编写而成，后来MATLAB 改用C 语言编写。自 1984 年由 MathWorks 公司正式把MATLAB 推向市场以来，MATLAB 就因其强大的数值运算和图形处理功能引起了科学计算和工程领域的广泛关注。MATLAB 作为一个多学科、多操纵平台的优秀科技软件，它自身也在不断地发展和进步，MATLAB 6.x 版本较以前版本的 MATLAB 变化较多，其处理运算的功能更加强大，各种领域的工具箱也日趋完善。现在，MATLAB 已经成为高等院校、科研院所以及广大工程技术人员首选的编程语言。

全书从 MATLAB 的基础知识入手，紧紧围绕应用程序实例，向读者展示了如何利用 MATLAB 实现符号运算功能，并详细介绍了 MATLAB 的符号函数绘图功能。本书图文并茂、实例众多，且所举出的实例针对性强，分析透彻，突出了本书以实例为中心的特点。本书是市场中为数不多的详细介绍 MATLAB 符号处理功能的工具书，相信通过阅读本书，会让读者对 MATLAB 产生浓厚的兴趣，加快 MATLAB 编程的速度，提高编程的技巧。

本书共分为 7 章。第 1 章介绍了 MATLAB 的安装与启动方法；第 2 章和第 3 章介绍了 MATLAB 的基础知识，这是 MATLAB 入门必备的知识；第 4 章介绍了 MATLAB 符号运算的初步知识；第 5 章和第 6 章介绍了 MATLAB 的符号处理功能，是本书的重点内容；第 7 章详细介绍了如何在 MATLAB 中绘制符号函数的图形；在附录中还介绍了怎样获取 MATLAB 符号运算函数及其命令的帮助信息。初学者最好按照本书的编写顺序系统地学习 MATLAB 语言，当然，已经用过 MATLAB 语言的读者，也可以挑选自己感兴趣的有关章节进行学习。不管是哪一类读者都会切身感受到本书的精练和实用。

本书由博嘉科技资讯有限公司组织编写。参加编写工作的还有胡晓闽、李莉、刘军、钱斌、吕可维、魏晓东、高柏松、曹阳、倪明、梁栩等，在此一并表示感谢。

由于 IT 产业发展日新月异，且 MATLAB 技术涉及面广泛，加之编写时间比较仓促，本书难免存在疏漏甚至错误，敬请广大读者批评指正。

编　者

# 目　　录

# 第1章　MATLAB的安装

**知识点：**

- MATLAB 的发展过程
- MATLAB 的安装及启动
- MATLAB 的在线帮助

本章主要介绍 MATLAB 的发展过程、MATLAB 的安装与启动以及 MATLAB 的在线帮助。由于 MATLAB 语言书写简便，因此 MATLAB 享有“演草纸”式语言的美誉。通过本章的学习，读者可以了解到 MATLAB 的发展历程，掌握 MATLAB 的安装方法，掌握进入 MATLAB 编程环境的两种不同途径。在掌握了本章介绍的 MATLAB 在线帮助技能以后，读者就可以在 MATLAB 编程环境中灵活使用 MATLAB 的在线帮助功能，从而大大提高编程能力。

## 1.1　MATLAB 的发展过程

MATLAB 从发展到现在已经经历了 20 多年的时间。它刚刚诞生时是各大学之间共享的一个免费软件包，后来发展成一个功能强大的软件系统。经过不断的版本升级，现在已经发展到 MATLAB 6.1 版。接下来，简要介绍一下 MATLAB 的发展历程。

MATLAB 是矩阵（Matrix）和实验室（Laboratory）两个英文单词的前三个字母的组合。它的首创者是美国新墨西哥大学计算机系的系主任 Cleve Moler 博士。他在教授线性代数课程时，发现用其他高级语言编程极为不便，便构思并开发了 MATLAB。这一软件利用了当时广为流行的 EISPACK（基于特征值计算的软件包）和 LINPACK（线性代数软件包）中的可靠子程序，利用 FORTRAN 语言编写的集命令翻译、科学计算于一体的交互式软件系统。20 世纪 80 年代初出现了 MATLAB 的第二代版本，该版本全部用 C 语言编写，使 MATLAB 不但具有数值计算功能，而且还具备了数据图示化功能。随着 MATLAB 应用范围的不断扩大，1984 年，Moler 博士和一批数学家及软件专家组建了一个名为 MathWorks 的软件开发公司，专门开发 MATLAB。1992 年，MathWorks 公司推出了具有划时代意义的 MATLAB 4.0 版本。1993 年，推出了 MATLAB 的微机版，可以和 Windows 联用，使之应用范围不断扩大。随着微软公司 Windows 9x 操作系统的推出，MathWorks 公司于 1997 年推出了基于 Windows 9x 操作系统的 MATLAB 5.0 版本。后来 MATLAB 又升级到 5.2 版本、5.3 版本。MATLAB 5.x 版本与以前的 4.x 版本相比，在界面与功能上都有了很大的进展。现在，MATLAB 已经升级到 6.1 版本，该版本的 MATLAB 运算功能得到进一步的扩充；对于图形的处理，可以用相应的编辑工具实现所见即所得；用户界面更为友好，也更符合用户的习惯；新增了与 Java 语言的接口；它还对绝大多数工具箱进行了功能扩充，使新老用户从中得到更大的

益处。

现在，MATLAB 已经被广泛应用于数值计算、图形处理、符号运算，数学建模、小波分析、系统辨识、实时控制和动态仿真等研究领域。在国外，MATLAB 已经成为大学生及研究生所必须掌握的一门计算机语言，并且成为科学研究、工程设计与运算等任务的得力助手。

## 1.2 MATLAB 的安装与启动

### 1.2.1 MATLAB 的安装

下面，以 MATLAB 6.1 的安装过程为例，说明如何安装 MATLAB 6.x。

安装 MATLAB 6.1 的过程主要有以下 5 个步骤：

步骤一：将 MATLAB 6.1 光盘放入光驱，然后双击名为 setup.exe 的文件，计算机开始进行 MATLAB 6.1 的安装初始化工作。首先将出现如图 1-1 所示的界面；紧接着出现如图 1-2 所示的版本信息界面；单击 Next 按钮，将出现如图 1-3 所示的用户权限界面。

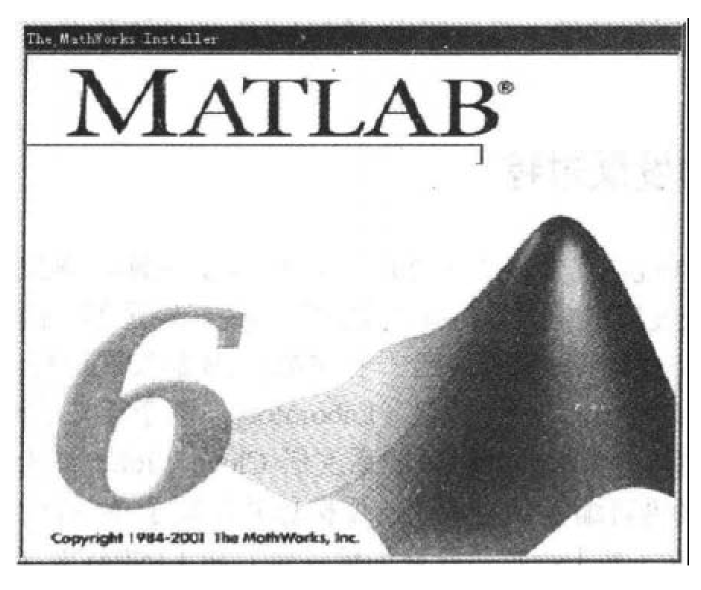



图 1-1 MATLAB 6.1 的安装界面

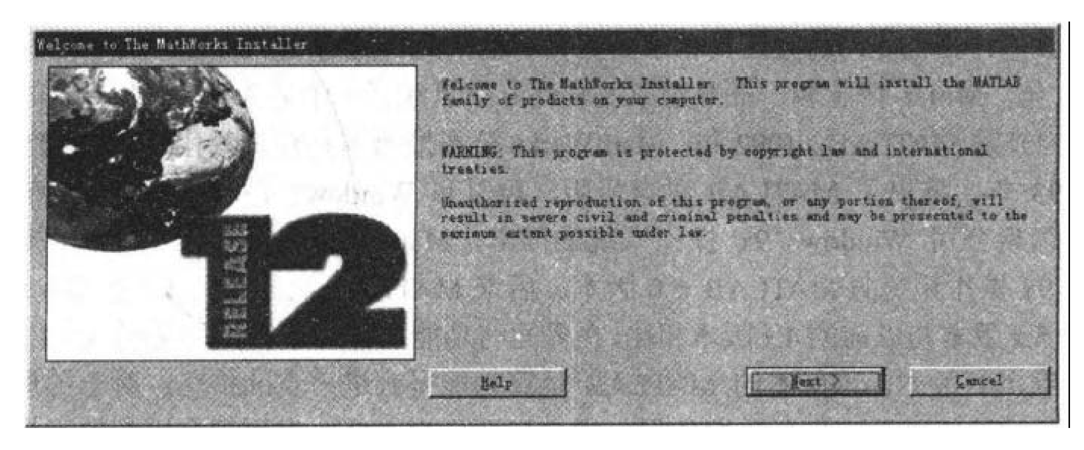



图 1-2 MATLAB 6.1 的版本信息界面

步骤二：在如图 1-3 所示的安装界面的对话框中输入安装密码，单击 Next 按钮则出现如图 1-4 所示的协议信息界面。单击 Yes 按钮（表示接受协议），则出现如图 1-5 所示的用户信息输入界面，填入用户名及公司名称以后，将出现如图 1-6 所示的选择安装界面。

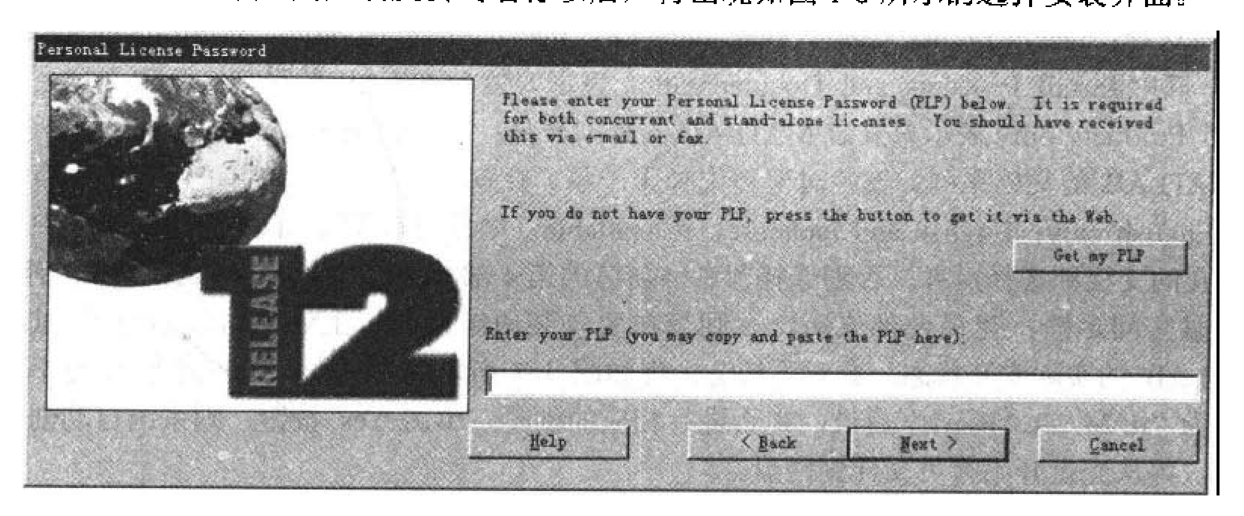



图 1-3　MATLAB 6.1 的用户权限界面

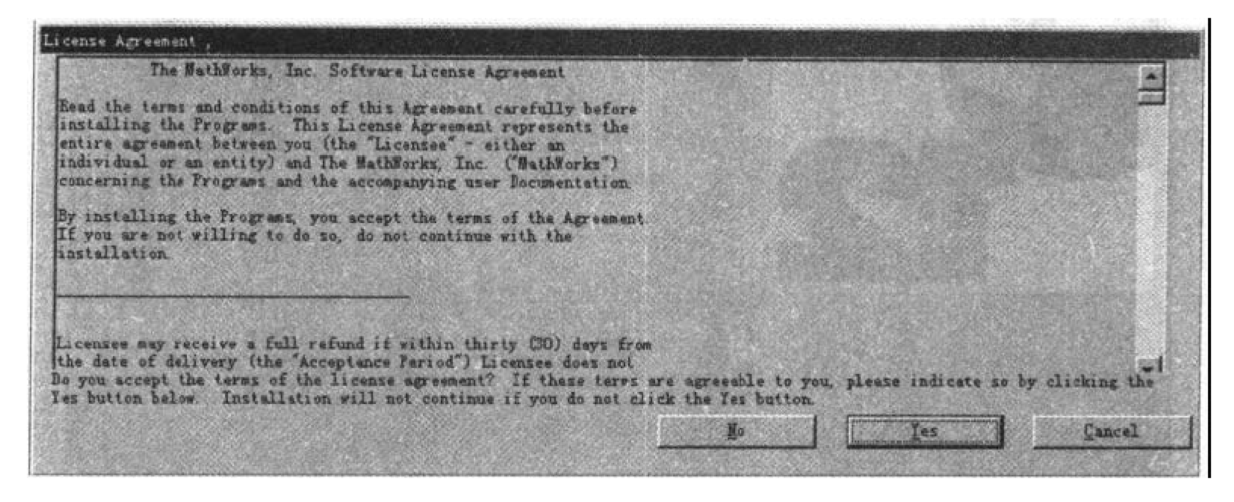



图 1-4　MATLAB 6.1 的协议信息界面

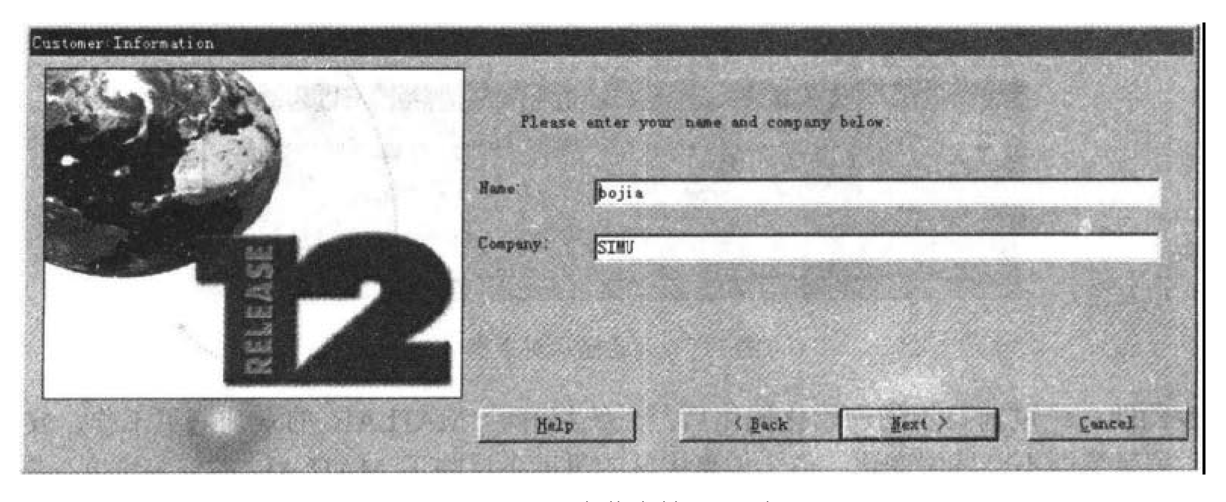



图 1-5　用户信息输入界面

步骤三：图 1-6 中的窗口分为 4 个部分，单击窗口第 1 部分中的 Browse 按钮可以改变 MATLAB 的安装路径。假如要把 MATLAB 6.1 安装到 D 盘的 MATLAB6p1 目录下，则只需要在图 1-6 中窗口的第 1 部分将“C:”改为“D:”即可。图 1-6 中窗口第 2 部分可进行不同的三种选择：Install Products and Documentation；Install Products only；Install documentation only。其中 Install Products and Documentation 表示安装 MATLAB 的系列产品和文档系列；Install Products only 表示只安装 MATLAB 的系列产品；Install documentation only 表示只安装 MATLAB 的文档系列。安装时建议选择第一条。在窗口的第 3 部分中可以选择文档的语种：English only；English and Japanese，if available。如果以后操作 MATLAB 的过程中要用到日文的文档则选择后者，否则选择前者。窗口的第 4 部分可以选择 MATLAB 的安装组件，若硬盘空间足够，建议全部安装，否则只在要安装的组件前打上√。单击 Next 按钮进入 MATLAB 的实际安装状态。此时系统会出现如图 1-7 所示的对话框，提示现在 d 盘不存在路径为“d:\MATLAB6p1”目录，是否进行相应的创建。单击 Yes 按钮，则系统自动创建该目录，并从光盘开始拷贝文件到硬盘。

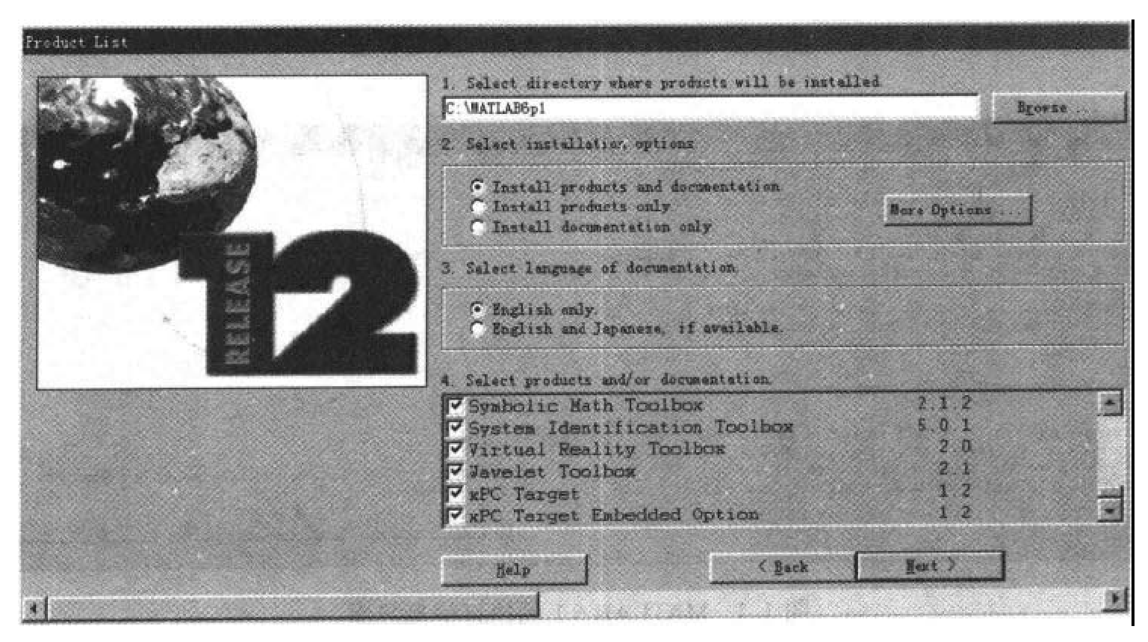



图 1-6　MATLAB 6.1 的选择安装界面

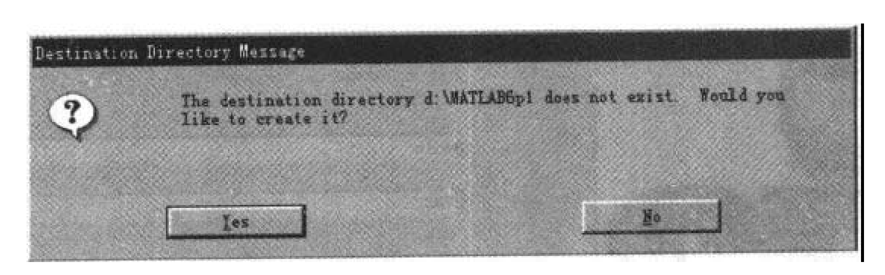



图 1-7　目录提示对话框

步骤四：经过前三步的初始化操作后，系统将开始 MATLAB 的安装拷贝工作。安装过程中，安装界面不断发生变化，位于屏幕中间的进度条反映了 MATLAB 的安装进度。图 1-8 是安装过程中出现的一个界面。

步骤五：安装到最后阶段将出现如图 1-9 所示的对话框，提示用户马上重新启动计算机

或者以后再启动计算机，默认的是前者，单击 Finish 按钮，则计算机将重新启动。至此为止，就完成了 MATLAB 6.1 的安装。

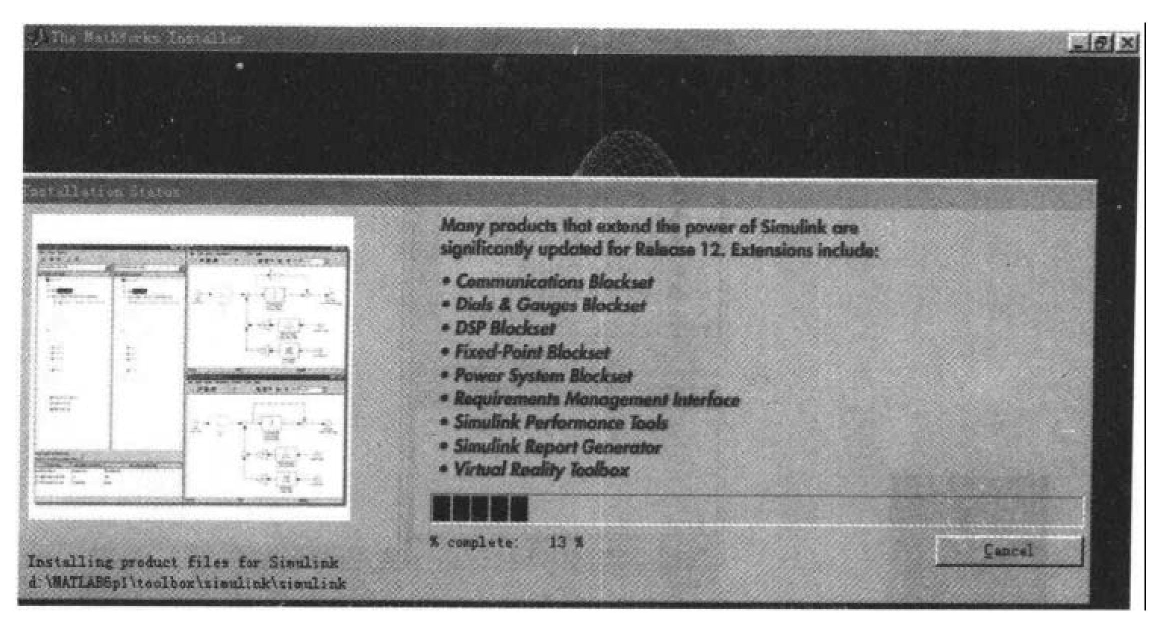



图 1-8 MATLAB 6.1 的安装进度界面

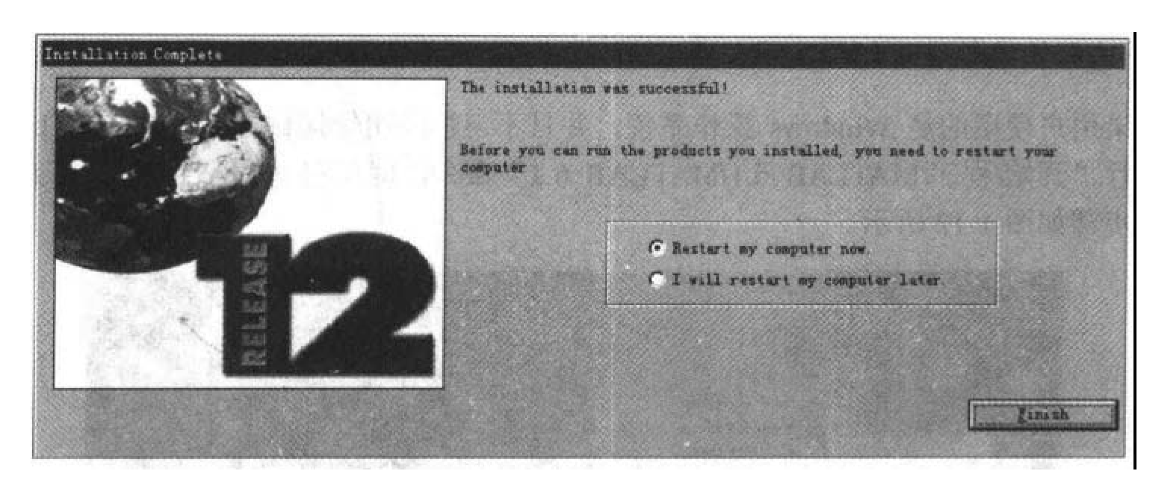



图 1-9 MATLAB 6.1 安装过程中的最终提示框

注意：MATLAB 6.0 的安装盘一般是 1 张光盘。MATLAB 6.1 的安装盘一般是 2 张光盘，其中，第一张盘用于安装 MATLAB 6.1 操作环境和各种工具箱，第二张盘是 MATLAB 6.1 的帮助文件安装盘，分为英文帮助（help）和日文帮助（jhelp）两个目录。若需要安装英文的帮助文件，则可以直接将 help 目录下的所有文件拷贝到 MATLAB 6.1 安装路径下的 help 子目录中。若需要安装日文帮助文件，也可以如法炮制。

## 1.2.2 MATLAB 的启动

启动 MATLAB 的常用方法有两种：利用快捷键启动以及由开始菜单启动。

### 1. 利用快捷键启动

MATLAB 安装完毕，重新启动计算机以后，会在桌面上自动建立启动 MATLAB 的快捷

方式图标，如图 1-10 所示。用鼠标双击该快捷方式就可以启动 MATLAB。MATLAB 启动后的界面如图 1-11 所示。




图 1-10 MATLAB 6.1 的快捷方式图标

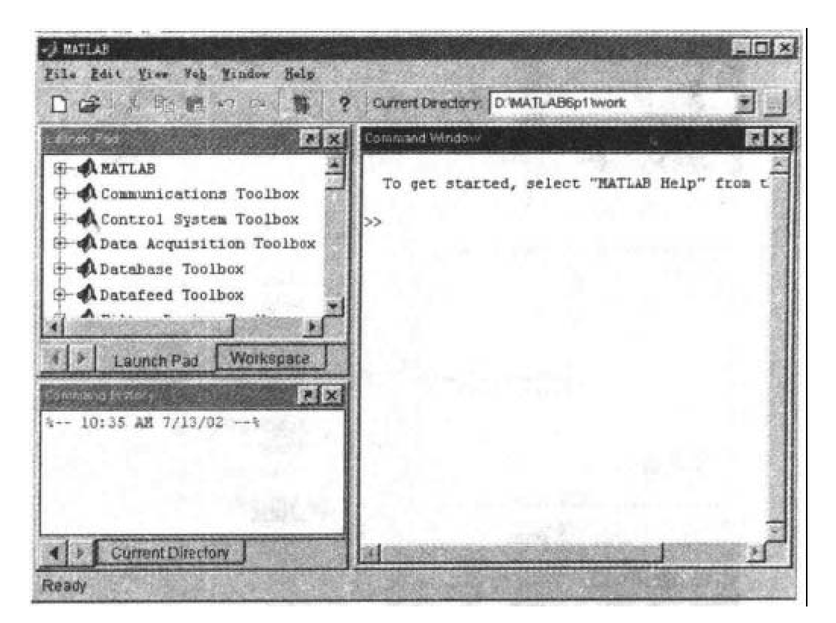



图 1-11 MATLAB 启动后的界面

### 2. 由开始菜单启动 MATLAB

如果用户使用的是 Windows 操作系统，并且不习惯使用快捷键方法启动 MATLAB，则可以执行“开始/程序/MATLAB 6.1/MATLAB 6.1”命令，即可启动 MATLAB 工作方式。上述操作步骤如图 1-12 所示。

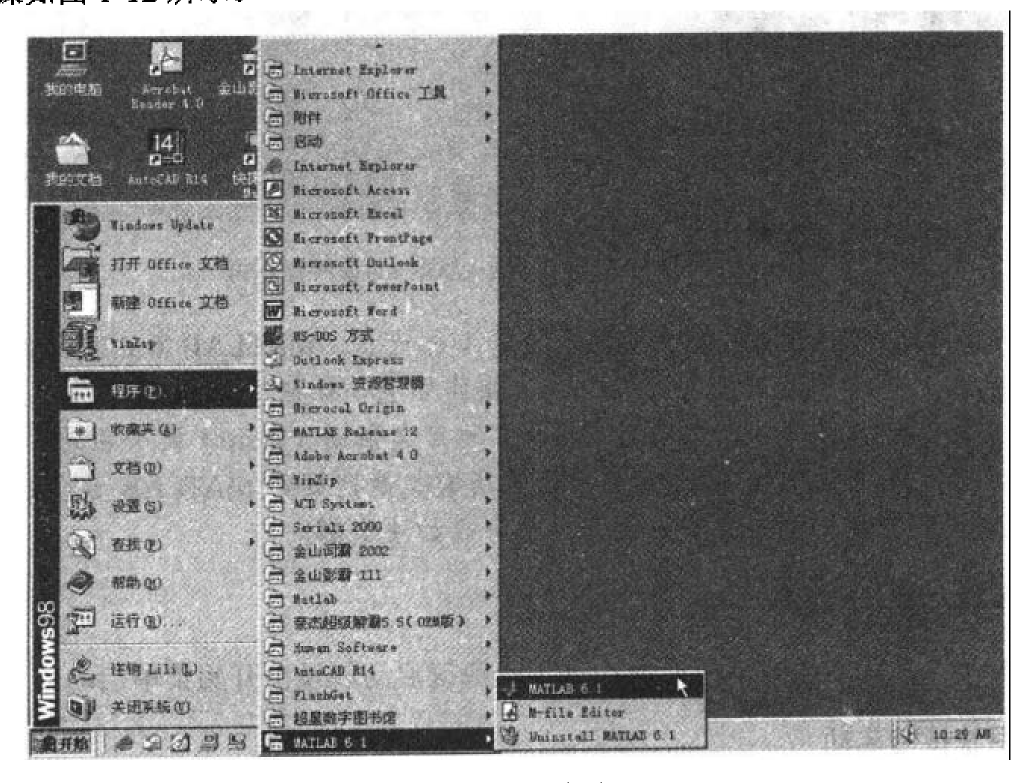



图 1-12 由开始菜单启动 MATLAB

## 1.3　MATLAB 的在线帮助

MATLAB 提供了丰富的查询帮助功能。总的来说，可以分为两大类：一类是在 MATLAB 的命令窗口通过输入命令来获得帮助，另一类是利用 MATLAB 提供的帮助浏览器。下面将分别进行介绍。

### 1.3.1　在 MATLAB 命令窗口获得帮助信息

在 MATLAB 的命令窗口中直接输入某些命令就可以获得相应的帮助信息。主要有以下几种命令：

#### 1．help 命令

其调用格式为：

- help：列出最原始的帮助项目，每项对应 MATLAB 的一个目录名。
- help func：根据用户指定的项目，func 给出相应的帮助信息。其中，func 可以是命令名、函数名也可以是目录名。如果是目录名，则 help 显示一指定目录的索引清单。

【例 1-1】

直接在 MATLAB 的命令窗口输入：

```
help
```

按 Enter 键后将显示以下信息：

```
HELP topics:
matlab\general      - General purpose commands.
matlab\ops          - Operators and special characters.
matlab\lang         - Programming language constructs.
matlab\elmat        - Elementary matrices and matrix manipulation.
matlab\elfun        - Elementary math functions.
matlab\specfun      - Specialized math functions.
matlab\matfun       - Matrix functions - numerical linear algebra.
matlab\datafun      - Data analysis and Fourier transforms.
matlab\audio        - Audio support.
matlab\polyfun      - Interpolation and polynomials.
matlab\funfun       - Function functions and ODE solvers.
matlab\sparfun      - Sparse matrices.
matlab\graph2d      - Two dimensional graphs.
matlab\graph3d      - Three dimensional graphs.
matlab\specgraph    - Specialized graphs.
matlab\graphics     - Handle Graphics.
matlab\uitools      - Graphical user interface tools.
matlab\strfun       - Character strings.
matlab\iofun        - File input/output.
matlab\timefun      - Time and dates.
```

```
    :                    :                     :                :
toolbox\rptgenext        -  Simulink Report Generator
toolbox\splines          -  Spline Toolbox.
toolbox\stats            -  Statistics Toolbox.
ident\ident              -  System Identification Toolbox.
ident\idobsolete            -  (No table of contents file)
ident\idguis             -  (No table of contents file)
ident\idutils            -  (No table of contents file)
ident\iddemos            -  (No table of contents file)
ident\idhelp             -  (No table of contents file)
vr\vr                       -  Virtual Reality Toolbox.
vr\vrdemos                  -  Virtual Reality Toolbox examples.
wavelet\wavelet             -  Wavelet Toolbox.
wavelet\wavedemo            -  Wavelet Toolbox Demonstrations.
xpc\xpc                     -  xPC Target
build\xpcblocks             -  (No table of contents file)
xpc\xpcdemos                -  xPC Target -- demos and sample script files.
kernel\embedded             -  xPC Target Embedded Option
MATLAB6P1\work              -  (No table of contents file)

For more help on directory/topic, type "help topic".
For command syntax information, type "help syntax".
```

【例 1-2】

查询 diff 命令的使用方法。在 MATLAB 命令窗口中输入并执行以下命令：

```
help diff
```

则显示结果为：

```
DIFF Difference and approximate derivative.
   DIFF(X), for a vector X, is [X(2)-X(1)  X(3)-X(2) ... X(n)-X(n-1)].
   DIFF(X), for a matrix X, is the matrix of row differences,
      [X(2:n,:) - X(1:n-1,:)].
   DIFF(X), for an N-D array X, is the difference along the first
      non-singleton dimension of X.
   DIFF(X,N) is the N-th order difference along the first non-singleton
      dimension (denote it by DIM). If N >= size(X,DIM), DIFF takes
      successive differences along the next non-singleton dimension.
   DIFF(X,N,DIM) is the Nth difference function along dimension DIM.
      If N >= size(X,DIM), DIFF returns an empty array.
    Examples:
      h = .001; x = 0:h:pi;
      diff(sin(x.^2))/h is an approximation to 2*cos(x.^2).*x
      diff((1:10).^2) is 3:2:19
      If X = [3 7 5
              0 9 2]
      then diff(X,1,1) is [-3 2 -3], diff(X,1,2) is [4 -2
```

```
                9 -7],
      diff(X,2,2) is the 2nd order difference along the dimension 2, and
      diff(X,3,2) is the empty matrix.
    See also GRADIENT, SUM, PROD.
Overloaded methods
   help sym/diff.m
   help char/diff.m
   help fints/diff.m
```

注意：MATLAB 是区分大小写的。虽然给出的帮助信息中有些 diff 是大写的，但是用户在使用过程中应当全部使用小写。

### 2. 其他查询命令

（1）lookfor

查找帮助信息中的关键字。

其调用格式为：

lookfor abc：系统按照已设置的 MATLAB 路径在所有文件中查找字符串 abc。对每个文件首部注释行的第一行所描述的帮助信息进行扫描。若在该行中找到字符串 abc，则将其所在的文件名以及所在行显示在屏幕上。

lookfor abc –all：与 lookfor abc 类似，但查找范围扩大为每个文件首部的第一个注释块。

【例 1-3】

若想获得关于命令 sin 的帮助信息，只需在 MATLAB 的命令窗口输入并执行以下命令：

```
lookfor sin
```

将显示以下帮助信息：

```
java.m: %Using Java from within MATLAB
syntax.m: %     You can enter MATLAB commands using either a FUNCTION format or a
SUBSINDEX Subscript index.
ISINF   True for infinite elements.
ACOS     Inverse cosine.
ACOSH    Inverse hyperbolic cosine.
ASIN     Inverse sine.
ASINH    Inverse hyperbolic sine.
COS      Cosine.
COSH     Hyperbolic cosine.
SIN      Sine.
SINH     Hyperbolic sine.
GSVD     Generalized Singular Value Decompostion.
SVD      Singular value decomposition.
DETREND Remove a linear trend from a vector, usually for FFT processing.
WAVPLAY Play sound using Windows audio output device.
WAVRECORD Record sound using Windows audio input device.
```

INTERPFT 1-D interpolation using FFT method.
EIGS Find a few eigenvalues and eigenvectors of a matrix using ARPACK.
SVDS Find a few singular values and vectors.
UILOAD Present file selection dialog and load result using LOAD
FITSINFO Information about FITS file
SINGLE Convert to single precision.
SQUEEZE Remove singleton dimensions.
CLEARCASE Version control actions using Clearcase.
PVCS Version control actions using PVCS.
RCS Version control actions using RCS.
SOURCESAFE Version control actions using Visual SourceSafe.
BURGERSODE Burger's equation solved using a moving mesh technique.
⋮ ⋮ ⋮ ⋮
BLKSINEGEN Generates a sine wave at a given time, start time, amplitude and frequency.
CHI2CONF Confidence interval using inverse of chi-square cdf.
dctold.m: %DCT Discrete cosine transform.
DIVIDENOWARN Divides two polynomials while suppressing warnings.
GENCOSWIN Returns one of the generalized cosine windows.
idctold.m: %IDCT Inverse discrete cosine transform.
INVSINC Desired amplitude and frequency response for invsinc filters.
DESIGNFILTER Designs a filter using the filter specifications stored
QUANTIZEAPPLY Action of pressing apply on quantize page.
render_spthelpmenu.m: %RENDER_HELP_MENU Render a Signal Processing Toolbox specific "Help" menu.
CSTRCHK Determines if argument is a vector cell of single line strings.
STRGLUE Concatenates a cell array of strings to a single string.

（2）which

查找指定函数和文件的目录。

其调用格式有：

- which fun：显示指定 fun 的全部路径和文件名。其中 fun 可以是工作空间的变量、内置函数、已经加载的 Simulink 模型或者是 java 类的方法。
- which fun –all：显示所有名为 fun 的函数路径。
- which file.ext：显示指定文件的全路径名称。
- which fun1 in fun2：显示在 m 文件 fun2 中出现的函数 fun1 的路径名称。
- which fun(a,b,c,...)：显示指定的带有输入参数 a，b，c，...的函数 fun 的路径。
- s=which(...)：把查询结果返回给字符串 s，而不是输出到屏幕上。当 fun 为内置函数时，s 的内容为'built-in'。
- w=which(...,'-all')：返回多项搜索方法的结果。w 是个细胞数组，它包含通常在屏幕上输出的路径字符串。

【例 1-4】

① 执行命令：

```
which sin
```

显示结果为：

```
is a built-in function.
```

说明 sin 命令是个内置函数。

② 执行命令：

```
which sin -all
```

显示结果为：

```
sin is a built-in function.
D:\MATLAB6P1\toolbox\symbolic\@sym\sin.m   % sym method
D:\MATLAB6P1\toolbox\matlab\elfun\sin.m     % Shadowed
```

③ 执行命令：

```
which solve in sin
```

显示结果为：

```
D:\MATLAB6P1\toolbox\symbolic\solve.m
```

表明 sin.m 中调用的 solve 函数位于目录 D:\MATLAB6P1\toolbox\symbolic\solve.m 中。

（3）who

列出当前工作空间中的变量。

【例 1-5】

在 MATLAB 的命令窗口输入：

```
who
```

按 Enter 键后将发现 MATLAB 的命令窗口没有发生什么变化，这表明还没有在 MATLAB 环境中创立变量。如果在 MATLAB 的命令窗口输入以下命令：

```
x=1,y=2
```

按 Enter 键后发现 MATLAB 的命令窗口变为：

```
x =
     1
y =
     2
```

此时，再在 MATLAB 的命令窗口输入：

```
who
```

按 Enter 键后发现 MATLAB 的命令窗口变为：

```
Your variables are:
x   y
```

（4）whos

列出当前工作空间中变量的详细信息。

【例 1-6】

在 MATLAB 的命令窗口输入：

```
whos
```

按 Enter 键后显示结果为：

```
  Name        Size                    Bytes  Class
  x           1x1                         8  double array
  y           1x1                         8  double array
Grand total is 2 elements using 16 bytes
```

说明此时 MATLAB 的当前工作空间中共有两个变量，并且给出了每个变量的名称、大小、字节数以及类型。

还可以通过以下方法观察有关变量的信息：执行 MATLAB 桌面上的“View/Desktop Layout/Five Panel”命令，此时，MATLAB 变为如图 1-13 所示的 5 面板结构形式。其中，Workspace 面板显示的就是 MATLAB 当前工作空间变量的有关信息。

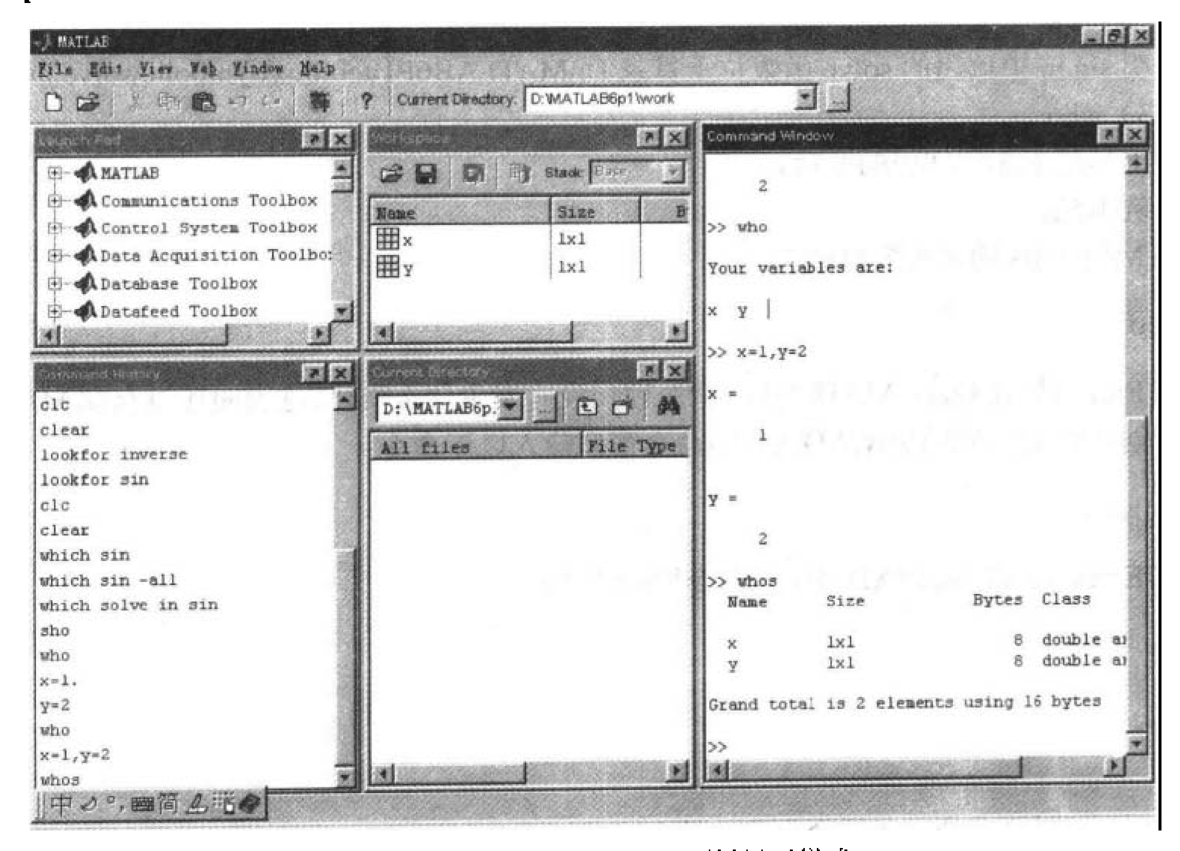



图 1-13　MATLAB 窗口的 5 面板显示样式

注意：使用 who global 以及 whos global 可以查询哪些变量是全局变量。

（5）exist

检查指定的变量名或者函数名是否存在。

调用格式为：

exist('a')

其返回值是下列9种情况中的一种：

- 0：表示a不存在。
- 1：表示a是工作空间中的变量。
- 2：表示a是MATLAB搜索路径下的m文件。如果a是文件的全目录形式或者a是一个普通文件名时，exitst('a')的结果也是2。
- 3：表示a是MATLAB搜索路径下的MEX文件。
- 4：表示a是MATLAB搜索路径下的MDL文件。
- 5：表示a是MATLAB的内置函数。
- 6：表示a是MATLAB搜索路径下的P文件。
- 7：表示a是目录。
- 8：表示a是java类。

（6）doc

在帮助浏览器中显示HTML文档。

doc命令有以下几种调用格式：

- doc：显示在线文档的起始页。
- doc function：显示MATLAB函数function的HTML文档。如果function已被重载，则在MATLAB的命令窗口列出重载的函数。
- doc toolbox/function：显示指定工具箱函数function的HTML文档。
- doc toolbox：显示指定产品的文档地图页。

【例1-7】

在MATLAB命令窗口输入：

```
doc
```

按Enter键后会出现如图1-14所示的Help窗口。用户可以在这个窗口中获得一系列的帮助信息。

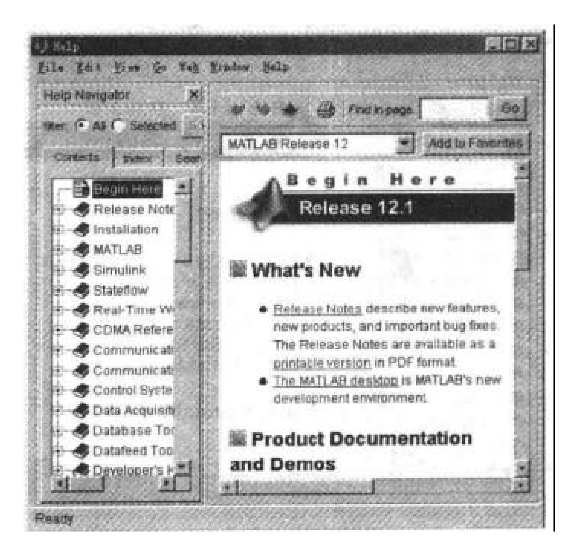



图1-14 Help窗口

【例 1-8】

在 MATLAB 命令窗口输入：

```
doc symbolic/solve
```

按 Enter 键后会出现如图 1-15 所示的 Help 窗口。由此可见，这种帮助方式给出的信息是相当全面的。

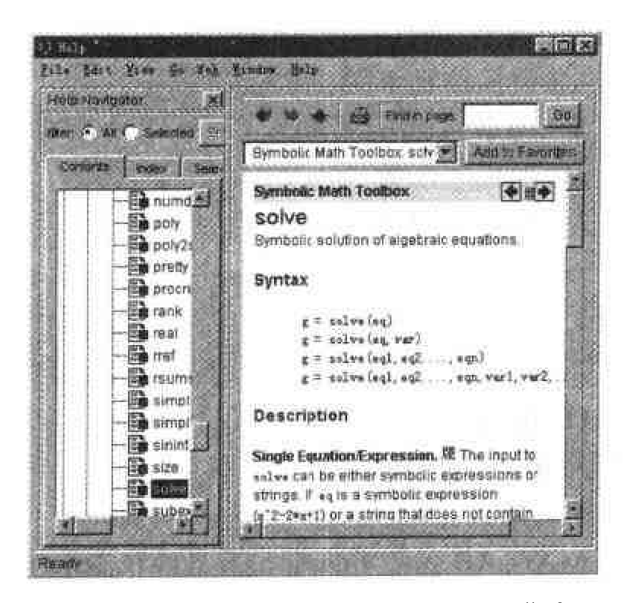


图 1-15　Help 窗口中出现的 solve 帮助信息

### 1.3.2　直接由帮助菜单获得帮助信息

单击 MATLAB 桌面菜单栏中的 Help 菜单，将出现以下可选项：

- Full Product Family Help：选择该项后可以获得 MATLAB 所有产品的帮助信息，如图 1-16 所示。



图 1-16　Help 菜单中与 Full Product Family 选项对应的帮助窗口

- Using Help：选择该项后可以获得有关 MATLAB 帮助浏览器的信息，如图 1-17 所示。

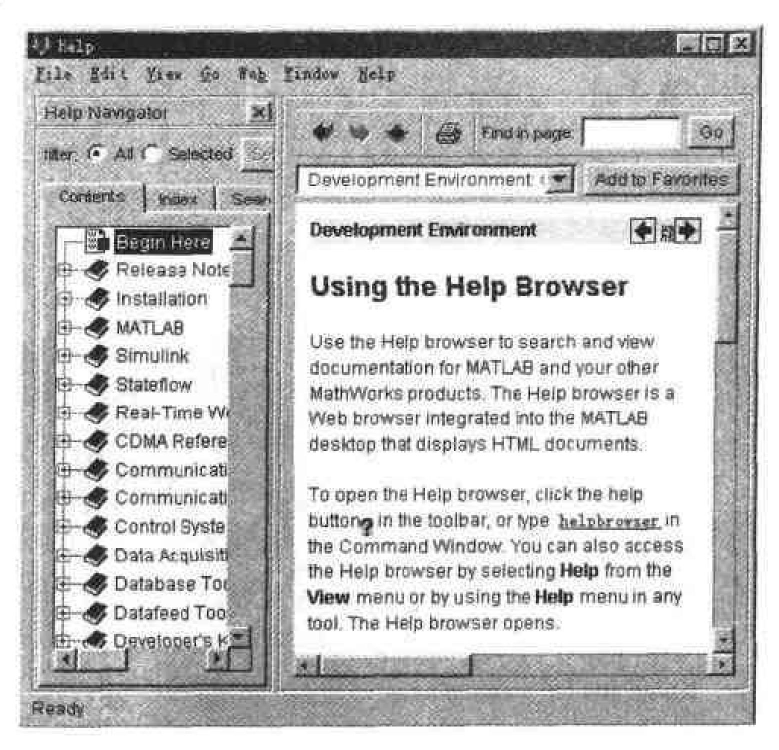



图 1-17　Help 菜单中与 Using Help 选项对应的帮助菜单

- Demos：执行 MATLAB 的演示功能，如图 1-18 所示。

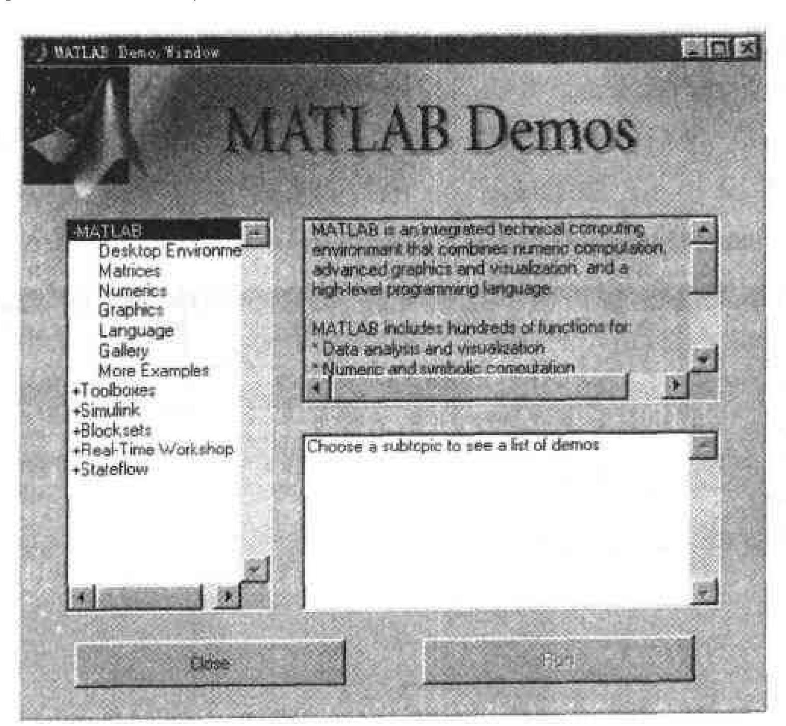



图 1-18　Help 菜单中与 Demos 选项对应的帮助菜单

- About MATLAB：用于查看关于 MATLAB 版本以及协议方面的信息，如图 1-19 所示。

【例 1-9】

试借助于 MATLAB 提供的帮助浏览器查找 sin 函数的有关帮助信息。

操作过程如下：

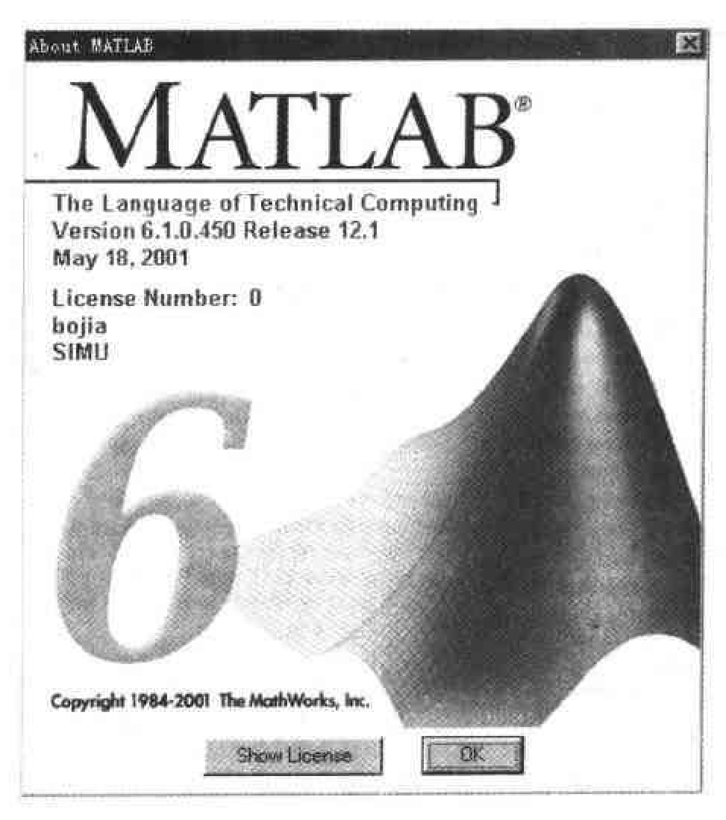



图 1-19　Help 菜单中与 About MATLAB 选项对应的帮助菜单

先在 MATLAB 桌面上的 Help 菜单中选中 MATLAB Help，这时将出现如图 1-17 所示的窗口。首先，在该窗口中单击 Search 制表键；其次，在 Search type 栏中选择 Function Name，在 Search for 栏中键入 sin；最后单击 Go 按钮，这时 MATLAB 的帮助浏览器将自动搜索 sin 函数的帮助信息，如图 1-20 所示。除了显示 sin 函数的帮助信息以外，还给出与该函数有关的其他函数的超级链接：asin, asinh。若用户想了解这些函数，则直接单击相应的选项就可以了。

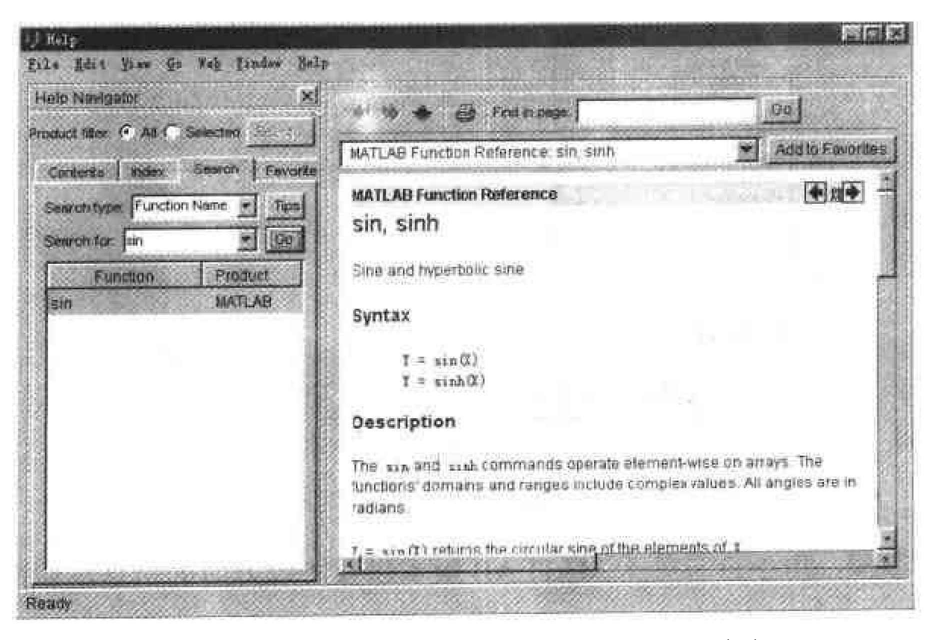



图 1-20　帮助浏览器中显示的 sin 函数的帮助信息

## 1.4 小结

MATLAB 的安装和启动都是比较简单的。作为初学者来说，关键要掌握 MATLAB 在线帮助的各种实现方式。需要注意的是，大而全的帮助显示方式并不一定是最好的显示方式，怎样才能恰当地选择符合自己习惯的帮助方式？只有在掌握 1.3 节有关内容的基础上，多学多练，熟能生巧，才会随心所欲地使用 MATLAB 的各种在线帮助功能。

# 第 2 章　MATLAB 语言的主要构成

知识点：

- 常量与变量
- 简单数组及其访问的实现
- 各种运算符

本章主要介绍 MATLAB 的基础知识。与其他计算机语言一样，MATLAB 语言的参变量也分为常量和变量两种形式。MATLAB 中所有的计算都是以矩阵为单位进行的，因此要熟练掌握 MATLAB 语言就必须首先从学习简单数组开始。MATLAB 可以实现各种运算，这些运算是借助于 MATLAB 提供的各种运算符实现的。通过本章的学习，读者可以达到基本了解 MATLAB 语言的目的。

## 2.1　常量与变量

MATLAB 中使用的参变量可分为常量和变量。例如，以下内容都是合法的 MATLAB 常量。

1　2.1　0.15*e*+2　5.2*e*+10　.5*e*+3　1+2.5*i*

注意：MATLAB 中的复数类型常量是可以直接写成手写格式的，这是与其他计算机语言的明显不同之处。

MATLAB 中实数的范围已经扩大到 $2.2251e^{-308}$~$1.7977e^{308}$，这样的取值范围对于绝大多数情况下的数值计算已经足够了。

MATLAB 中变量的命名方式与其他语言的变量命名方式基本相似，如要求以字母打头，由字母、数字或下划线组成。一般来说，变量名的长度不要超过 31 个字符。若不同变量的变量名前面 31 个字符是相等的，则 MATLAB 认为它们是相同的变量。

注意：MATLAB 的文件名不受字符长度的限制。只要用户愿意，即使是长度多达几百个字符的文件名，MATLAB 也一样能识别。

MATLAB 使用的变量不必事先定义，用户完全可以采取现场定义的方式来定义变量。变量名最好不要随便命名，最好能达到言简意赅的目的。另外，MATLAB 已经预定义了一些特殊变量，如表 2-1 所示。

表 2-1 MATLAB 中的特殊变量

| MATLAB 预先定义的特殊变量 | 含 义 |
| --- | --- |
| ans | 最近生成的默认变量值 |
| clock | 时钟 |
| computer | 返回运行 MATLAB 的机型 |
| cputime | CPU 的运行时间 |
| date | 当前的日期信息 |
| datenum | 日期的数字序列格式 |
| datestr | 日期的字符串格式 |
| datevec | 日期的矢量格式 |
| eomday | 月的最后一天 |
| eps | 浮点数相对误差，为 $2.2204e^{-016}$ |
| etime | 得到计算机的运行时间 |
| i,j | 虚数单位 |
| Inf | 无穷大（如 1/0） |
| inputname | 输入参数的名称 |
| NaN | 不定值（如 0.0/0.0） |
| nargin | 函数输入参数的个数 |
| nargout | 函数输出参数的个数 |
| now | 当前日期和时间 |
| pi | 圆周率 |
| realmax | 最大正浮点数（为 $1.7977e^{308}$） |
| realmin | 最小正浮点数（为 $2.2251e^{-308}$） |
| tic | 启动计时秒表 |
| toc | 读取秒表的运行时间 |
| version | MATLAB 的相关版本信息 |
| weekday | 周日 |

## 2.2 简单数组及其访问的实现

### 2.2.1 简单数组简介

MATLAB 中的数组包括：行矢量（可视为 1 行多列的数组）、列矢量（可视为 1 列多行的数组）以及矩阵（可视为多行多列的数组）。本节只举一个简单的例子使读者明白 MATLAB 是如何用数组来解决问题的，关于数组的各种运算参见 2.3 节内容。

比如，已知函数 $y=x^3$，其中 $x\in[0,2]$区间。现在要在[0,2]区间上每隔 0.2 取一个值计算其平方值。这在 MATLAB 中是很容易实现的：首先创建一个数组 $x$，其相邻两个元素的差值为 0.2，然后就可以直接计算对应点的 y 值了。

在 MATLAB 的命令窗口中，执行以下命令：

```
x=0:0.2:2
```

则得到的执行结果为：

```
x =
  Columns 1 through 7
         0    0.2000    0.4000    0.6000    0.8000    1.0000    1.2000
  Columns 8 through 11
    1.4000    1.6000    1.8000    2.0000
```

再执行命令：

```
y=x.^3
```

则得到的执行结果为：

```
y =
  Columns 1 through 7
         0    0.0080    0.0640    0.2160    0.5120    1.0000    1.7280
  Columns 8 through 11
    2.7440    4.0960    5.8320    8.0000
```

由此可见，在 MATLAB 中实现上述任务是非常方便的，同时用户还可以在 MATLAB 桌面的命令记录窗口（Command History 窗口）看到 MATLAB 已经运行过的命令列表。若用户使用了 MATLAB 桌面的 5 面板显示格式，那么用户可以在工作空间窗口（Workspace 窗口）直接看到已经建立的变量 *x*,*y*。若用户想删除变量 *x*，既可以在命令窗口中键入命令 clear *x* 实现，也可以先将 MATLAB 桌面设置成 5 面板显示格式，在其中的工作空间窗口中用鼠标选中变量 *x*，然后通过使用键盘上的 Delete 键，或右击鼠标从弹出的菜单中选择 Delete Selection 选项把 *x* 删除。在删除的过程中，MATLAB 会自动弹出一个确认对话框，询问用户是否确信将所选的变量从工作空间中删除。

☞ 提示：上述命令 *x*=0:0.2:2 中的赋值格式是 MATLAB 常用的变量赋值格式，其中 0 表示初始值，0.2 表示增量，而 2 表示终止值。若数组 *x* 没有规律可循，那么为 x 赋值只好逐一输入单个元素了，这时要使用赋值格式符“[]”，如 *x*=[0 5 8 1]表示把元素 0 5 8 1 赋予变量 *x*。

命令 *y*=*x*.^3 中的“.”表示对数组 *x* 中的元素进行操作而不是把数组看成一个整体进行操作，显然求一个矢量的立方是没有意义的（数组 *A*,*B* 相乘必须满足 *A* 的列数与 *B* 的行数相等）。若用户输入命令 *y*=*x*^3 则系统会给出出错信息。命令 *y*=*x*.^3 中的“^”表示数组的幂指数即对数组进行相乘的操作。

### 2.2.2 数组的建立

通常可以采用两种方法来创建数组：逐个元素输入法和冒号法。在这里介绍另外两种常用的数组创建方法：linspace 法和 logspace 法。在理论研究和工程实践中经常要用到的线性等距的数组以及对数等距的数组，在 MATLAB 中可以分别用命令 linspace 和 logspace 实现。为了说明问题的方便，假设要建立起始值为 0、终止值为π、间隔为 0.1π的数组 *x*，那么用前 3 种方法创建矢量 *x* 的具体命令分别为：

```
x=[0 0.1*pi 0.2*pi 0.3*pi 0.4*pi 0.5*pi 0.6*pi 0.7*pi 08*pi 0.9*pi pi]    %逐个元素赋值法
x=(0:0.1:1)*pi          %冒号赋值法，也可以写成   x=0:0.1*pi:pi
x=linspace(0,pi,10)     %linspace 赋值法，表示将 0~pi 分成 10 个等间距点。
```

从上述命令可以看出，若数组的元素有规律可循，则一般不用逐个元素赋值的办法创建数组，因为该种方法是最为繁琐的一种创建方法。

如想在区间[0,100]上得到对数间距相等的 10 个点，此时使用 logspace 命令是最简单的，具体执行命令为：

```
x=logspace(1,2,10)
```

linspace 和 logspace 命令的具体调用方法相似：

- linspace(*x*1,*x*2)：得到一个长度为 100 的行矢量，它是通过把[*x*1,*x*2]区间分成间距相等的 100 个点生成的。
- linspace(*x*1,*x*2,*N*)：在区间[*x*1,*x*2]之间产生 *N* 个点，所得结果为一个行矢量。
- logspace(*d*1,*d*2)：在区间[$10^{d1}$,$10^{d2}$]之间产生对数间距相等的 50 个点，返回结果为一个行矢量。若 *d2=pi* 则产生的点在 $10^{d1}$~*pi* 之间。
- logspace(*d*1,*d*2,*N*)：在区间[$10^{d1}$,$10^{d2}$]之间产生对数间距相等的 *N* 个点，返回结果为一个行矢量。同样，当 *d2=pi* 时，产生的点位于 $10^{d1}$ ~*pi* 之间。

【例 2-1】

（1）分别执行以下命令：

```
x1=linspace(1,10)
x2=linspace(1,10,5)
x3=logspace(-1,1)
x4=logspace(-1,1,10)
```

则 x2,x4 的显示结果为：

```
x2 =
     1.0000     3.2500     5.5000     7.7500    10.0000
x4 =
   Columns 1 through 7
     0.1000     0.1668     0.2783     0.4642     0.7743     1.2915     2.1544
   Columns 8 through 10
     3.5938     5.9948    10.0000
```

其中，*x*1,*x*3 的元素太多，这里不再列出显示结果，读者可自己上机实践。

（2）执行命令：

```
logspace(0,pi,5)
```

则执行结果为：

```
ans =
     1.0000     1.3313     1.7725     2.3597     3.1416
```

此时产生的点位于 $10^0$ 和π之间，而不是位于 $10^0$ 和 $10^\pi$之间。另外，从上面的执行结果

可以看出，在不指明赋值变量名的情况下，MATLAB 把变量名全部置为特定变量 ans。

### 2.2.3 数组的访问

MATLAB 用圆括号来表示数组的下标，访问数组中的元素也是使用圆括号，与其他计算机语言不同的是：MATLAB 中 $x(n)$表示的是数组 $x$ 的第 $n$ 个元素。在圆括号中使用冒号则可以访问数组中的多个元素。

【例 2-2】

在 2.2.2 一节例 2-1 中已经创建了变量 $x2$，如执行以下命令：

```
x2(2)    %访问 x2 中的第 2 个元素
```

则得到的执行结果为：

```
ans =
    3.2500
```

如执行以下命令：

```
x2(1:3)   %访问 x2 中的前 3 个元素
```

则得到的执行结果为：

```
ans =
    1.0000    3.2500    5.5000
```

如执行以下命令：

```
x2(1:2:5)   %访问 x2 的第 1、第 3、第 5 个元素
```

则执行结果为：

```
ans =
    1.0000    5.5000   10.0000
```

如执行以下命令：

```
x2(5:-2:1)   %访问 x2 的第 5、第 3、第 1 个元素
```

则执行结果为：

```
ans =
   10.0000    5.5000    1.0000
```

当然，也可以用已经定义的数组作为参数来访问数组，如执行以下命令：

```
a=[1 3 4]   %定义数组 a
x2(a)       %访问 x2 的第 1、3、4 个元素
```

则得到的执行结果为：

```
a =
     1     3     4
ans =
    1.0000    5.5000    7.7500
```

☞ 提示：在上面的命令行中，百分号后面的内容是对语句的说明，主要是为了便于读者阅读。MATLAB 执行到百分号时，就把百分号后面的内容作为注释行对待，而不执行。

## 2.3 各种运算符

MATLAB 具有强大的计算功能，既能进行一般的标量计算，也能以整个矩阵为单位进行计算；它既可以处理一般的数值计算，也可以进行关系运算和逻辑运算。本节将简要介绍 MATLAB 中的各种运算符。

### 2.3.1 一般运算符

一般运算符和操作符构成了 MATLAB 最基本的操作命令。在本小节中将介绍 MATLAB 中的一般运算符，对于操作符的介绍将在 2.3.2 节中进行。

在 MATLAB 中几乎所有的计算都是以矩阵作为基本运算单元进行计算（因为标量和矢量都可以看成特殊的矩阵），这一点是 MATLAB 语言与其他语言的不同之处，也是 MATLAB 语言本身的一大特色，用户应当时刻注意到这一点。

MATLAB 中最常见的运算符如表 2-2 所示。下面简单介绍各项命令，以使读者对 MATLAB 的常规计算有一个大致的了解。

表 2-2　MATLAB 中最常见的运算符

| 运　算　符 | 功　能 |
|---|---|
| + | 加法 |
| - | 减法 |
| * | 矩阵乘法 |
| .* | 数组乘法 |
| ^ | 矩阵乘方 |
| .^ | 数组乘方 |
| / | 左除 |
| \ | 右除 |
| .\ | 数组左除 |
| ./ | 数组右除 |
| kron | Kronecker 张量积 |
| : | 冒号运算符 |
| ' | 共轭转置符 |
| .' | 一般转置符 |
| = | 赋值号 |
| () | 小括号，用于决定计算顺序，也用于数组的访问 |
| [] | 中括号，用于生成数组和矩阵 |

（续）

| 运 算 符 | 功 能 |
| --- | --- |
| {} | 大括号，用于生成单元数组 |
| . | 小数点或访问结构的域 |
| @ | 创建函数句柄 |

**1．矩阵的加减运算（+、-）**

一般来说，两个矩阵能进行加减运算的前提是它们必须具有相同的维数。两个矩阵进行加减运算时，两个矩阵的对应元素进行加减运算，所得的结果仍然是一个矩阵。作为特例，当一个标量和一个矩阵相加减时，MATLAB 就把这个标量和矩阵中的所有元素进行加减运算，当然所得结果也是一个矩阵。

【例 2-3】

首先，分别执行以下命令：

```
A=[1 2 3
   4 5 6];
B=[10 20 30
   40 50 60];
a=5;
```

接下来，如执行以下命令：

```
C1=A+B
```

则得到的执行结果为：

```
C1 =
   11    22    33
   44    55    66
```

如执行以下命令：

```
C2=a+A
```

则得到的执行结果为：

```
C2 =
    6     7     8
    9    10    11
```

如执行以下命令：

```
C3=A-B
```

则得到的执行结果为：

```
C3 =
   -9   -18   -27
```

```
-36    -45    -54
```

☞ 提示：MATLAB 中若在一个表达式的后面写上分号“;”，则 MATLAB 计算完该表达式后并不显示表达式的计算结果（但该计算结果已经保存到 MATLAB 的工作空间中，只是在 MATLAB 的命令窗口看不到得到的执行结果，而在 MATLAB 的工作空间窗口中还是可以看到该计算结果所赋予的变量名，只是不知道结果是多少，此时在命令窗口直接输入该变量回车后就会显示出该变量的实际值是多少）；若不在表达式的末尾加分号，那么 MATLAB 将显示表达式的计算结果。另外，参加加减运算的矩阵还可以是复数阵，如例 2-4 所示。

【例 2-4】

分别执行以下命令：

```
A=[1  2  3; 4  5  6];                %A 为实数阵
B=[1+i  0  2-i;3  -5-i  5-2i];       %B 为复数阵
a=2;                                 %a 为标量
C1=A-B                               %计算 A-B 的值
C2=B+a                               %计算 B+a 的值
C3=a-B                               %计算 a-B 的值
```

则执行结果分别为：

```
C1 =
        0 - 1.0000i     2.0000              1.0000 + 1.0000i
   1.0000              10.0000 + 1.0000i    1.0000 + 2.0000i
C2 =
   3.0000 + 1.0000i    2.0000               4.0000 - 1.0000i
   5.0000              -3.0000 - 1.0000i    7.0000 - 2.0000i
C3 =
   1.0000 - 1.0000i    2.0000                    0 + 1.0000i
  -1.0000              7.0000 + 1.0000i    -3.0000 + 2.0000i
```

☞ 提示：例 2-4 中给矩阵 A、B 赋值时，在中括号之内使用了分号，表示 MATLAB 执行到该处，将开始矩阵的下一行输入。

**2．矩阵的乘法运算（*）**

在线性代数中：两个矩阵 A、B 能够进行相乘运算的前提条件是 A 的列数要与 B 的行数相等，若 A 是 m×n 的矩阵，B 是 n×p 的矩阵，则所得的结果是一个 m×p 的矩阵。作为退化情况，若其中有一个矩阵变成标量时，此时的运算规则是：该元素和矩阵中的所有元素进行乘法运算，所得结果仍是一个矩阵。

【例 2-5】

分别执行以下命令：

```
A=[10  20 ;30  40]
B=[1  2  3; 4  5  6]
a=2;
```

```
C1=A*B
C2=A*a
C3=a*B
```

则得到的执行结果为：

```
A =
    10    20
    30    40
B =
     1     2     3
     4     5     6
C1 =
    90   120   150
   190   260   330
C2 =
    20    40
    60    80
C3 =
     2     4     6
     8    10    12
```

### 3. 矩阵的数组乘法（.*）

A、B 两个矩阵进行数组相乘应满足的条件是：A 与 B 具有相同的维数。数组相乘时，其运算规则是两个矩阵的对应元素相乘，得到的结果是与 A、B 维数相同的矩阵。当有一个矩阵退化为一个标量时，A.*B 与 A*B 的结果是一样的。

【例 2-6】

分别执行以下命令：

```
A=[1   2;3   4];
B=[5   6;7   8];
a=2;
C1=A.*B
C21=a.*A
C22=a*A
C3=B.*a
```

则得到的执行结果分别为：

```
C1 =
     5    12
    21    32
C21 =
     2     4
     6     8
C22 =
     2     4
```

```
      6     8
C3 =
     10    12
     14    16
```

注意：矩阵乘法和矩阵的数组乘法是两个不同的概念，后者是矩阵的对应元素相乘。

**4. 矩阵的乘方（^）**

MATLAB 在计算表达式 A^B 时，其处理规则如下：

若 A 为方阵，B 为大于 1 的整数，则所得结果是 A 连乘 B 次所得到的结果；如果 B 不是整数，则计算各特征值和特征矢量的乘方。

若 A 为数而 B 为方阵，则计算 A^B 时，其结果也由特征值和特征矢量计算得到。

若 A 和 B 都是方阵，或者两个矩阵中有一个不是方阵，则 MATLAB 会给出出错信息。

【例 2-7】

分别执行以下命令：

```
A=[1 2;3 4];
B=[1 2 3;4 5 6];
C11=A^2
C12=A*A
C2=A^0.5
C3=1.2^A
C4=B^2
```

则执行结果分别为：

```
C11 =
     7    10
    15    22
C12 =
     7    10
    15    22
C2 =
   0.5537 + 0.4644i   0.8070 - 0.2124i
   1.2104 - 0.3186i   1.7641 + 0.1458i
C3 =
    1.3473    0.6019
    0.9028    2.2501
??? Error using ==> ^
Matrix must be square.
```

由计算结果 C11 与 C12 相等可知，对方阵 A 来说，A^2 和 A*A 的效果是一样的，都是计算矩阵 A 的平方。从计算结果也可得知，当矩阵不是方阵时，无法对它进行乘方运算。

### 5. 矩阵的数组乘方（.^）

A.^B 运算的前提是矩阵 A 与 B 具有相同的维数，当其中一个是数时，则对另外一个矩阵的所有元素执行乘方运算。

【例 2-8】

分别执行以下命令：

```
A=[1 2;3 4];
B=[2 3;4 5];
C=[1 2 3;4 5 6];
Y1=A.^B
Y2=B.^A
Y3=2.^A
Y4=A.^0.5
Y5=A.^C
```

则得到的执行结果为：

```
Y1 =
            1          8
           81       1024
Y2 =
     2     9
    64   625
Y3 =
     2     4
     8    16
Y4 =
    1.0000    1.4142
    1.7321    2.0000
??? Error using ==> .^
Matrix dimensions must agree.
```

### 6. 矩阵的左除（\）

对矩阵 A 和 B 来说，A\B 表示矩阵 A 左除矩阵 B，其计算结果与 A 的逆和 B 相乘相似，但 MATLAB 所用的算法不同。事实上，A\B 是方程 AX=B 的解，当方程是欠定或超定时，A\B 对应的是最小二乘解。

【例 2-9】

执行以下程序：

```
A=[1 2;3 4];
B=[3;5];
X=A\B
```

则得到的执行结果为：

```
X =
    -1
     2
```

事实上，上述结果就是 AX=B，即 $\begin{cases} x_1 + 2x_2 = 3 \\ 3x_1 + 4x_2 = 5 \end{cases}$ 的解。

**7．矩阵的右除（/）**

对矩阵 A 和 B 来说，A/B 称为矩阵 A 右除矩阵 B，其计算结果与 B 和 A 的逆相乘相似，但所用算法不同。事实上，它是方程 XA=B 的解。

【例 2-10】

执行以下命令：

```
A=[1 2;3 4];
B=[3 5;7 9];
X=B/A
```

则得到的执行结果为：

```
X =
    1.5000    0.5000
   -0.5000    2.5000
```

显然，X 就是 XA=B 的解。

**8．矩阵的点除（.\，./）**

- A./B：若 A 和 B 是维数相同的矩阵，则 A./B 就是 A 中的元素除以 B 中的对应元素，所得结果仍是一个矩阵，其维数与 A、B 的维数相同。如果 A、B 中有一个退化为标量，则结果为该标量和相应矩阵中的所有元素作运算，结果也是一个矩阵。
- A.\B：若 A 和 B 是维数相同的矩阵，则 A.\B 就是 B 中的元素除以 A 中的对应元素，所得结果仍是一个矩阵，其维数与 A、B 的维数相同。如果 A、B 中有一个退化为标量，则结果为该标量和相应矩阵中的所有元素作运算，结果也是一个矩阵。

【例 2-11】

分别执行以下命令：

```
A=[1 2;3 4];
B=[10 20;30 40];
X1=A./B
X2=A.\B
X3=A./2
X4=2./A
```

得到的执行结果分别为：

```
X1 =
    0.1000    0.1000
```

```
    0.1000    0.1000
X2 =
    10    10
    10    10
X3 =
    0.5000    1.0000
    1.5000    2.0000
X4 =
    2.0000    1.0000
    0.6667    0.5000
```

### 2.3.2 操作符

本节将介绍 MATLAB 中常用的几个操作符。

1.：（冒号）

冒号在 MATLAB 中的应用十分广泛，它既可用作矩阵的下标，又可以应用于行循环操作等。

- *j:k*：等价于[*j*  *j*+1  *j*+2 … *k*]，若 *j*>*k* 则返回空值。
- *j*: *i*:*k*：[*j* *j*+*i*  *j*+2*i* … *k*]，若 *j*>*k* 且 *i* >0 或者 *j*<*k* 且 *i* <0，则返回空值。
- *A*(:,*i*)：取矩阵 *A* 的第 *i* 列，其中冒号表示取矩阵的所有行。
- *A*(*i*,:)：取矩阵 *A* 的第 *i* 行，其中冒号表示取矩阵的所有列。
- *A*(*i*:*i*)：以矩阵 *A* 的所有元素构造一个二维矩阵，若 *A* 本身就是二维矩阵，则结果不变。
- *A*(:)：将 *A* 的所有元素作为一个列矢量，如果此操作符在赋值语句的左边，则用右边矩阵的元素来填充 *A*。矩阵 *A* 的结果不变，但要求两边矩阵元素的个数要相等，否则会给出出错信息。

*A*(*j*:*k*)：等价于 *A*(*j*),*A*(*j*+1),…,*A*(*j*+*k*)。

【例 2-12】

分别执行以下命令：

```
A=[1 2 3 4;5 6 7 8;9 10 11 12]    %A 是一个 3×4 的矩阵
B=rand(2,2,3)                     %B 是一个 2×2×3 的随机三维矩阵
Va1=A(:,2)                        %该命令把 A 的第 2 列赋予 Va1
Va2=A(2,:)                        %该命令把 A 的第 2 行赋予 Va2
Va3=A(:)                          %该命令把 A 中的元素作为一个列矢量赋予 Va3
VB1=B(:,:)                        %以 B 的所有元素构造一个二维矩阵 VB1
A(:)=B                            %用 B 中的所有元素填充 A
```

则得到的执行结果分别为：

```
A =
    1    2    3    4
    5    6    7    8
```

```
     9     10     11     12
B(:,:,1) =
     0.9501    0.6068
     0.2311    0.4860
B(:,:,2) =
     0.8913    0.4565
     0.7621    0.0185
B(:,:,3) =
     0.8214    0.6154
     0.4447    0.7919
Va1 =
      2
      6
     10
Va2 =
      5      6      7      8
Va3 =
      1
      5
      9
      2
      6
     10
      3
      7
     11
      4
      8
     12
VB1 =
     0.9501    0.6068    0.8913    0.4565    0.8214    0.6154
     0.2311    0.4860    0.7621    0.0185    0.4447    0.7919
A =
     0.9501    0.4860    0.4565    0.4447
     0.2311    0.8913    0.0185    0.6154
     0.6068    0.7621    0.8214    0.7919
```

读者可仔细分析以上执行结果，从而找出矩阵的冒号操作规律。

### 2. %（百分号）

前面已经讲过，MATLAB 中的%（百分号）当作注释符，在一行的某处键入%，则该行%以后的部分将被视为注释部分而不再执行。

本书在讲解 MATLAB 的使用过程中，很多地方都大量使用了注释符，以便于读者理解某些命令的执行目的。

另外，在编程过程中给程序的主要部分加上注释行（以利于程序的阅读）是个很好的习

惯，希望读者在日常编程中养成这一习惯。

百分号用在 MATLAB 的函数中则另有用途。例如，编写一个求两个元素之积的函数，命名为 multiply，程序清单为：

```
function y=multiply (a,b)
%This function can compute the multiply of two elements,
%The input arguments a,b are scarlars

%   Begin
      y=a*b;  %get the result
```

在 MATLAB 命令窗口中输入：

```
help multiply
```

则可以获得函数 multiply 的有关帮助信息：

```
This function can compute the multiply of two elements,
The input arguments a,b are scarlars
```

由此可见，紧靠 function y= multiply (a,b)以下、带有%的注释行可以利用 help multiply 命令显示出来，但是一旦有空行，则后面的注释行通过使用 help 命令是显示不出来的。

**3. ...（三个连续的点）**

MATLAB 中把三个连续的点作为续行标志。尽管现在 MATLAB 已经取消了命令行长度的限制，即用户可以在一行中输入很多条命令，但是有时为了书写的美观，不愿意把一个程序的所有命令都写成一行。若确实有超过一屏宽度的命令行时，则可以在适当的地方加入续行标志，然后另起一行继续书写命令。

**4. '（撇号）**

表示矩阵的转置。

【例 2-13】

如转置矩阵 $A=\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\end{bmatrix}$，在 MATLAB 中可以执行以下命令：

```
A=[1 2 3;4 5 6];
B=A'
```

则得到的执行结果为：

```
B =
      1      4
      2      5
      3      6
```

**5. ;（分号）**

前面已经讲过，分号用在每行命令的结尾，则 MATLAB 执行完该行命令时不会显示本行的计算结果。当分号用在中括号“[]”内时，表示矩阵中该行的结束。

【例 2-14】

在 MATLAB 中执行以下命令：

```
A=[1 2 3;4 5 6]
```

则得到的执行结果为：

```
A =
     1     2     3
     4     5     6
```

### 2.3.3 关系运算符

MATLAB 提供的关系运算符包括：<（小于）、<=（小于等于）、>（大于）、>=（大于等于）、==（等于）、~=（不等于）。

**1. <（小于）**

*A*<*B* 的结果是这样给出的：如果 *A* 矩阵的元素小于 *B* 矩阵中对应位置的元素，则在结果矩阵的相应位置上输出 1，否则输出 0；如果其中之一是标量，则将这个标量与另一矩阵的每一个元素进行比较，结果也是一个包含元素 0 和 1 的矩阵。

另外，函数 lt(*A*,*B*)也能完成 *A*<*B* 的判断，其中 *A*、*B* 可以是矩阵、数值或其他任意成员（如 figure 对象）。

【例 2-15】

若矩阵 $A=\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$，矩阵 $B=\begin{bmatrix}0 & 2\\3 & 5\end{bmatrix}$，分别执行以下命令：

```
A=[1 2;3 4];
B=[0 2;3 5];
a=2;
C11=A<B
C12=lt(A,B)
C2=a<A
```

则得到的执行结果为：

```
C11 =
     0     0
     0     1
C12 =
     0     0
     0     1
```

```
C2 =
     0     0
     1     1
```

### 2. <=（小于等于）

*A*<=*B*，若 *A* 矩阵中的元素小于或者等于 *B* 矩阵中与之相对应的元素，则在结果矩阵的相应位置上输出 1，否则输出 0；如果其中一个是标量，则将这个标量与另一矩阵的每个元素进行比较，所得结果仍为矩阵。

另外，函数 le(*A*,*B*)也能完成 *A*<=*B* 的判断，其中 *A*、*B* 可以是矩阵、数值或其他任意成员（如 figure 对象）。

【例 2-16】

矩阵 *A*、*B* 以及标量 *a* 与例 2-15 中给出的数值均相同，分别执行以下命令：

```
A=[1 2;3 4];
B=[0 2;3 5];
a=2;
C11=A<=B
C12=le(A,B)
C2=a<=A
```

则得到的执行结果为：

```
C11 =
     0     1
     1     1
C12 =
     0     1
     1     1
C2 =
     0     1
     1     1
```

### 3. >（大于）

*A*>*B*，与 *A*<*B* 的计算结果正好相反，这里不再赘述。

函数 *gt*(*A*,*B*)也能完成 *A*>*B* 的判断。

【例 2-17】

矩阵 *A*、*B* 以及标量 *a* 与例 2-15 中给出的数值均相同，分别执行以下命令：

```
A=[1 2;3 4];
B=[0 2;3 5];
a=2;
C11=A>B
C12=gt(A,B)
C2=a>A
```

则得到的执行结果为：

```
C11 =
     1     0
     0     0
C12 =
     1     0
     0     0
C2 =
     1     0
     0     0
```

### 4. >=（大于等于）

*A*>=*B*，运算方法与<=相似，判断 *A* 的元素是否不小于 *B* 中对应位置的元素，具体判断方法在此不再赘述。

函数 *ge*(*A*,*B*)也能完成 *A*>=*B* 的判断。

【例 2-18】

分别执行以下命令：

```
A=[5 4;3 2];
B=[5 3;1 4];
a=3.5;
C1=A>=B
C2=ge(A,B)
C3=a>=B
```

得到的执行结果为：

```
C1 =
     1     1
     1     0
C2 =
     1     1
     1     0
C3 =
     0     1
     1     0
```

### 5. ==（等于）

*A*==*B*，当 *A* 和 *B* 是同维的矩阵时，若 *A* 矩阵中的元素与 *B* 矩阵中对应位置的元素相等，则在结果矩阵中输出 1，否则输出 0。若 *A*、*B* 之一是标量时，则该标量与另一矩阵中的所有元素进行比较。若相等就在结果矩阵中输出 1，否则输出 0。

函数 *eq*(*A*,*B*)也能完成 *A*==*B* 的判断，这里的 *A*、*B* 可以是任意的矩阵、数值或其他任意对象（如 figure）。

【例 2-19】

执行以下命令：

```
A=[1 2;3 4];
B=[2 2;2 3];
a=2;
C1=A==B
C2=eq(A,B)
C3=a==B
```

则得到的执行结果为：

```
C1 =
      0     1
      0     0
C2 =
      0     1
      0     0
C3 =
      1     1
      1     0
```

### 6. ~=（不等于）

*A~=B*，判断 *A* 中的元素是否不等于对应位置 *B* 中的元素，算法与==相似，在此不再赘述。

函数 *ne*(*A*,*B*)也能完成 *A~=B* 的判断，同样，该函数中的 *A* 和 *B* 可以是任意的矩阵、数值或者其他成员。

【例 2-20】

矩阵 *A*、*B* 以及标量 *a* 与例 13 中给出的数值均相同，则执行以下命令：

```
A=[1 2;3 4];
B=[2 2;2 3];
a=2;
C1=A~=B
C2=ne(A,B)
C3=a~=B
```

则得到的执行结果为：

```
C1 =
      1     0
      1     1
C2 =
      1     0
      1     1
C3 =
      0     0
      0     1
```

## 2.3.4 逻辑运算符与逻辑函数

### 1. 逻辑运算符

MATLAB 提供的逻辑运算符包括：&（与）、|（或）、~（非）和 xor（异或）。

其中，表达式 *a*|*b* 和 xor(*a*,*b*)的区别是：当 *a*、*b* 中只要其中之一为真时，那么经过 *a*|*b* 运算后就为真。显然当 *a*、*b* 均为真时，*a*|*b* 的结果也为真。而 xor(*a*,*b*)则是只有当 *a*、*b* 之一为真时，经过 xor(*a*,*b*)计算后结果才为真。显然当 *a* 和 *b* 均为真时，xor(*a*,*b*)的计算结果为假。

MATLAB 在给出逻辑运算的结果时以数值“1”代表真，以“0”代表假，但在判断一个量是否为真时，以任意的非 0 值代表真，以 0 代表假。另外，逻辑运算符是按元素进行比较的，参与逻辑运算的两个对象可以都是矩阵，也可以都是标量，还可以其中之一是矩阵，另一个为标量。当其中之一为标量时，将该标量与另一矩阵的所有元素进行逻辑运算，返回结果仍是由 0 和 1 组成的矩阵，并且该矩阵和参加运算的矩阵具有相同的维数。

- *A*&*B*：返回一个与 *A*、*B* 具有相同维数的矩阵。在这个矩阵中，当 *A*、*B* 对应位置的元素都是非 0 值时，则结果矩阵中对应位置输出 1，如果有一个为 0，则结果矩阵中该位置输出 0。
- *A*|*B*：返回一个与 *A*、*B* 具有相同维数的矩阵。在这个矩阵中，当 *A*、*B* 对应位置的元素只要有一个为非 0 值时，则结果矩阵中对应位置为 1。当两个元素均为 0 时，则在结果矩阵中对应位置为元素 0。
- ~*A*：返回与 *A* 具有相同维数的矩阵。*A* 中某位置元素为 0，则在结果矩阵中该位置元素为 1；*A* 中某位置元素为非 0 值时，则在结果矩阵中该位置元素为 0。
- xor(*A*,*B*)：返回一个与 *A*、*B* 具有相同维数的矩阵。在这个矩阵中，如果 *A* 和 *B* 中同一个位置元素均为 0 或均为非 0 时，则结果矩阵中对应位置的元素为 0，除了这种情况则对应位置为 1。

为使读者对逻辑运算有一个直观的了解，表 2-3 给出了逻辑运算的计算规则。

表 2-3 逻辑运算的规则

| *A*(*i*,*j*) | *B*(*i*,*j*) | *A*&*B*（与） | *A*\|*B*（或） | ~*A*（非） | xor(*A*,*B*)（异或） |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 |

☞ 提示：其中，*A*(*i*,*j*)表示处于矩阵 *A* 的第 *i* 行、第 *j* 列的元素，*B*(*i*,*j*)依次类推。

注意：在一个表达式中，算术运算符优先级最高，其次是关系运算符，逻辑运算符的级别最低。在逻辑运算符中，“非”的优先级最高。MATLAB 6.x 中逻辑“与”（AND 或&）比逻辑“或”（OR 或者|）的优先级要高。另外，用圆括号则可以改变优先级的顺序。

【例 2-21】

执行以下程序：

```
A=[1 2;3 4];
B=[0 3;1 0];
a=2;
C1=A&B
C2=a&A
C3=A|B
C4=a|B
C5=~A
C6=xor(A,B)
```

则得到的执行结果：

```
C1 =
     0     1
     1     0
C2 =
     1     1
     1     1
C3 =
     1     1
     1     1
C4 =
     1     1
     1     1
C5 =
     0     0
     0     0
C6 =
     1     0
     0     1
```

在 MATLAB 中，也可以使用函数 and(*A*,*B*)，or(*A*,*B*)和 not(*A*,*B*)分别实现 *A*&*B*，*A*|*B* 和~*A*。

### 2．逻辑函数

MATLAB 提供的逻辑函数可以方便地查找矩阵中满足某些特定条件的元素或所有元素。这一点，其他语言是不能与之相比的。读者应当灵活掌握 MATLAB 提供的逻辑函数。

### 3．all 函数

功能：用于判断是否所有的元素均为非 0 值。

调用格式有：

- all(*x*)：其中 *x* 为矢量，若 *x* 中所有的元素均为非 0 值则返回 1，否则返回 0。
- all(*A*)：若 *A* 为矩阵，则对 *A* 的列进行操作。如果某列的所有元素均为非 0 值，则该

列的返回值为 1，因此对于矩阵 *A*，执行命令 all(*A*)其返回结果是一个矢量。

- all(*A*,dim)：在 dim 维内进行 all(*A*)的计算，即将指定的第 dim 维作为矢量进行计算。对二维的矩阵来说，all(*A*,1)则是对 *A* 的列进行判断。

【例 2-22】

执行以下命令：

```
A=[1 2 3
   4 0 6];
B=rand(1,2,3)           %矩阵 B 为 1×2×3 的三维列矢量
a1=all(A)               %A 为 2×3 的矩阵，此时 al(A)对 A 的列进行判断，
                            %因此结果必须是一个长度为 3 的行矢量
a2=all(A>=1)            %逻辑函数可以和逻辑操作符连用。
                            %该命令判断 A 中每列的元素是否都大于等于 1
a3=all(all(A>=0))       %all 命令的嵌套使用
                            %对由 all(A>=0)得到的矢量进行判断，故结果必为标量
a4=all(B)                   %all 命令可以判断多维矩阵
```

则得到的执行结果为：

```
B(:,:,1) =
    0.9218    0.7382
B(:,:,2) =
    0.1763    0.4057
B(:,:,3) =
    0.9355    0.9169
a1 =
     1     0     1
a2 =
     1     0     1
a3 =
     1
a4(:,:,1) =
     1
a4(:,:,2) =
     1
a4(:,:,3) =
     1
```

☞ 提示：由例 2-22 可以看出，三维列矢量在执行 all 命令时，是将第 2 维作为矢量进行逻辑判断的。

在编程过程中应当充分利用 all 函数的功能，对矢量或矩阵进行操作并根据返回的结果进行相应的处理。例如：

```
if all(all(A>=0.1))
    ......
    ......
```

```
    ......
end
```

### 4. any 函数

功能：判断是否有一个矢量元素为非 0 值。

在矩阵处理中，经常遇到要判断矩阵中的元素是否有 0 值存在的情况（例如，在对矩阵进行数组除运算时往往要先判断作除数的矩阵是否有 0 元素存在），MATLAB 提供的 any 函数就可以实现这一功能。

调用格式有：

- any(*x*)：其中 *x* 为矢量，用于对矢量 *x* 进行判断。如果矢量中至少有一个元素为非 0 值，则 any(*x*)的返回结果为真（即返回 1）。只有当 *x* 的所有元素均为 0 时，any(*x*)的返回结果才为 0。
- any(*A*)：若 *A* 为矩阵，则对 *A* 的列进行操作。如果某列存在某个元素为非 0 值，则该列的返回值为 1。只有当该列中的所有元素均为 0 时，对应该列的返回结果才为 0，因此 any(A)的结果是一个由 1 和 0 组成的矢量。

any(*A*,dim)：将指定的第 dim 维作为矢量进行计算。

【例 2-23】

执行以下命令：

```
A=[1 0 2
   1 0 0];
test0=any(A)
test1=any(A,1)
test2=any(A,2)
```

则得到的执行结果为：

```
test0 =
     1     0     1
test1 =
     1     0     1
test2 =
     1
     1
```

由计算结果可知，对二维的矩阵 *A*，命令 any(*A*)和 any(*A*,1)都是对 *A* 的列矢量进行判断，而命令 any(*A*,2)则对 *A* 的行矢量进行判断。

### 5. exist 函数

功能：查看变量或者函数是否存在。

当想查看某个变量是否已被定义过时，可以使用 MATLAB 提供的 exist 命令来完成这一任务。其调用格式为：

- a=exist('*A*')：该命令的返回值 *a* 与 *A* 对应的状态如表 2-4 所示。

表 2-4　命令 exist('A ')与返回结果 a 的对应关系

| a 的值 | A 的状态 |
| --- | --- |
| 0 | A 不存在 |
| 1 | A 是工作空间中的变量 |
| 2 | A 是 m 文件或是在 MATLAB 搜索路径下的未知类型的文件 |
| 3 | A 是 MATLAB 搜索路径下的 MEX 文件 |
| 4 | A 是 MATLAB 搜索路径下的已编译的 Simulink 函数（MDL 文件） |
| 5 | A 是 MATLAB 的内置函数 |
| 6 | A 是 MATLAB 搜索路径下的 A 文件 |
| 7 | A 是路径，但不一定是 MATLAB 搜索路径 |
| 8 | A 是一个 Java 类 |

如果对象 *A* 存在于 MATLAB 的搜索路径下，但并不是 MATLAB 可以识别的非 m 文件（即非 mdl 文件，非 p 文件，非 mex 文件）时，则 exist('*A* ')或者 exist ('A.ext')将返回 2。另外，命令 exist('*A* ')中，*A* 可以是当前路径下的子目录或相对路径。

- exist('*A*','var')：只检查工作空间中变量 A 是否存在。
- exist('*A*', 'builtin')：只检查内置函数 A 是否存在。
- exist('*A*', 'file')：检查文件 A 或者目录 A 是否存在。
- exist('*A*', 'dir')：只检查目录 A 是否存在。
- exist('*A*', 'class')：检查类 A 是否存在。

【例 2-24】

（1）已知 filtdes.p 是 toolbox\signal 目录下的一个 p 文件。如执行以下命令：

```
exist('filtdes')
```

则得到的执行结果为：

```
ans =
     6
```

（2）如执行以下命令：

```
exist('sin')   % sin 是 MATLAB 的内置函数
```

则得到的执行结果为：

```
ans =
     5
```

（3）已知 MATLAB 的当前路径是 D:\MATLAB6p1，如执行以下命令：

```
exist('bin')
```

则得到的执行结果为：

```
ans =
```

```
     7
```

（4）已知D盘存在名为fjl的目录，执行以下命令：

```
exist('d:\fjl')
```

则得到的执行结果为：

```
ans =
     7
```

（5）假如已经在工作空间中创建了变量 $a$，则输入命令：

```
exist('a','var')
```

得到的执行结果为：

```
ans =
     0
```

（6）搜索D盘是否存在名为turbo的目录，则输入命令：

```
exist('d:\turbo','dir')
```

得到的执行结果为：

```
ans =
     0
```

### 6. find 函数

功能：查找非0元素的下标。

find 命令有以下几种调用格式：

- find($x$)：其中 $x$ 为矢量。返回值是一个矢量，该矢量的元素由矢量 $x$ 中非0元素的下标组成。如果矢量 $x$ 中的所有元素都为0，则返回空阵。
- find($A$)：其中 $A$ 为矩阵。返回值为一矢量，表示 $A$ 中非0元素的下标。
- [$I$,$J$]=find($A$)：其中 $A$ 为矩阵。返回矢量 $I$ 和 $J$，为 $A$ 中非0元素的下标，即 $A$ 中元素 $(I_k,J_k)$ 为非0值。
- [$I$,$J$,$W$]=find($A$)：其中 $A$ 为矩阵。同上，但还返回矢量 $W$，表示 $A$ 中元素 $(I_k,J_k)$ 为非0元素并且能在 $W_k$ 中找到。

【例 2-25】

（1）执行以下命令：

```
x=[0 1 3 0 4];
find(x)
```

则得到的执行结果为：

```
ans =
     2     3     5
```

（2）执行以下命令：

```
y=[0 0 0];
find(y)
```

则得到的执行结果为：

```
ans =
     []
```

由此可见，对空矢量执行 find 命令后将返回空阵。

（3）执行以下命令：

```
X=[1 0 3;4 5 0;0 6 7];
find(X)
```

则得到的执行结果为：

```
ans =
     1
     2
     5
     6
     7
     9
```

由此可见，find 命令对二维矩阵的操作是按照列进行的。

（4）执行命令：

```
A(:,:,1)=[10 0 30;0 40 60];
A(:,:,2)=[20 0 0;0 70 0];
find(A)
[I,J,W]=find(A)
```

则得到的执行结果为：

```
ans =
     1
     4
     5
     6
     7
    10
I =
     1
     2
     1
     2
     1
     2
```

```
J =
      1
      2
      3
      3
      4
      5
W =
     10
     40
     30
     60
     20
     70
```

由该例可以看出，对于多维数组，其元素是按照从低维到高维依次进行标识的。例如，对于上面的多维数组 A，分别对应的下标为：

$$A(:,:,1)=\begin{bmatrix} A_1 & A_3 & A_3 \\ A_2 & A_4 & A_6 \end{bmatrix},\quad A(:,:,2)=\begin{bmatrix} A_7 & A_9 & A_{11} \\ A_8 & A_{10} & A_{12} \end{bmatrix}$$

巧妙地利用 find 命令可以对矩阵进行各种处理，如例 2-26 所示。

【例 2-26】

已知矩阵 $A=\begin{bmatrix} 0 & 1 & 2 & 4 \\ 3 & 0 & 0 & 3 \\ 4 & 7 & 8 & 0 \end{bmatrix}$，完成以下任务：

任务一：试将矩阵 $A$ 中的元素 0 全部换成-5，并把结果返回给矩阵 B。

任务二：将矩阵 $A$ 中为 3 的元素全部换成 0，并把结果返回给矩阵 C。

实现上述任务的命令分别为：

（1）完成任务一，可以执行以下命令：

```
A=[0 1 2 4;3 0 0 3;4 7 8 0];
A(find(A==0))=-5;
B=A
```

则得到的执行结果为：

```
B =
   -5     1     2     4
    3    -5    -5     3
    4     7     8    -5
```

（2）完成任务二，可以执行以下命令：

```
A=[0 1 2 4;3 0 0 3;4 7 8 0];
A(find(A==3))=0;
```

```
C=A
```

则得到的执行结果为：

```
C =
     0     1     2     4
     0     0     0     0
     4     7     8     0
```

### 7．isempty 函数

功能：确认矩阵是否为空矩阵。

isempty 命令的调用格式为：

● isempty(*A*)：若矩阵 *A* 为空阵则返回 1，否则返回 0。

【例 2-27】

执行以下命令：

```
A=[];
a=isempty(A)
```

则得到的执行结果为：

```
a =
      1
```

这时，在 MATLAB 的工作空间（Workspace）那一栏可以看到以下结果：

```
A              0x0      0       double array
```

表明 *A* 是一个 0×0 的双精度类型的空矩阵。

### 8．isequal 函数

功能：确认矩阵是否相等。

isequal 命令的调用格式为：

● isequal(*A*,*B*)：若矩阵 *A*、*B* 完全相同则返回 1，否则返回 0。

● isequal(*A*,*B*,*C*,...)：若矩阵 *A*、*B*、*C*、...完全，相同则返回 1，否则返回 0。

【例 2-28】

执行以下命令：

```
A=[1 2;3 4];
B=[1 2 3 4];
C=[1 2
    3 4];
eAB=isequal(A,B)
eAC=isequal(A,C)
eCB=isequal(C,B)
```

则执行结果为：

```
eAB =
```

```
        0
eAC =
        1
eCB =
        0
```

### 9. isnumeric 函数

功能：确认矩阵是否为数值型。

isnumeric 命令的调用格式为：

- isnumeric(*A*)：如果 *A* 是数值型矩阵（不管是双精度型实数矩阵还是复数阵）则该命令返回 1，否则返回 0。

【例 2-29】

分别执行以下命令：

```
a=[1 2 0];
A=[1 2;3+2i 0];
B=['as' 'sd'; 'df' 'fg'];
isa=isnumeric(a)
isA=isnumeric(A)
isB=isnumeric(B)
```

得到的执行结果分别为：

```
isa =
        1
isA =
        1
isB =
        0
```

☞ 提示：稀疏矩阵和双精度数组都是数值型的，而字符串、单元数组和结构数组都是非数值型的。

### 10. isa 函数

功能：检测给定的 MATLAB 类或 Java 类的对象。

调用格式为：

K=isa(obj,'class_name')：参数 obj 是 MATLAB 对象或 Java 对象，参数 class_name 是 MATLAB（预定义的或用户自定义的）或 Java 的类名。如果 obj 是 class_name 的类或者是它的子类则返回 1，否则返回 0。

预定的 MATLAB 类包括：

- cell：单元数组。
- char：字符型数组。
- double：双精度浮点型数组。
- function_handle：函数句柄。

- int8：带符号的 8 位（bit）整数数组。
- int16：带符号的 16 位（bit）整数数组。
- int32：带符号的 32 位（bit）整数数组。
- numeric：整数或浮点型数组。
- single：单精度浮点型数组。
- sparse：二维实型（或复数型）稀疏矩阵。
- struct：结构数组。
- unit16：无符号的 16 位整数数组。
- unit32：无符号的 32 位整数数组。

关于逻辑函数就讲解这么多，读者记住以下逻辑函数也是大有裨益的：

- ischar(A)：如果 A 是字符型数组则返回 1，否则返回 0。
- isfield(A)：如果 A 是某个结构体的域则返回 1，否则返回 0。
- ishandle(A)：如果 A 是图形句柄则返回 1，否则返回 0。
- islogical(A)：如果 A 是逻辑数组则返回 1，否则返回 0。
- isnan(A)：返回一个维数与 A 相同的数组。在这个数组中，对应 A 中有 NaN 的地方为 1，其他地方为 0。
- isreal(A)：如果 A 是一个不带虚部的实型数组则返回 1，否则返回 0。
- issparse(A)：判断 A 中存储的类是否是稀疏的，如果 A 是稀疏的则返回 1，反之则返回 0。
- isstruct(A)：如果 A 是结构体则返回 1，否则返回 0。

## 2.4　小结

与其他计算机语言一样，常量和变量是必须掌握的两个概念，MATLAB 已经预定义的常量一定要认真掌握。与其他计算机语言明显不同的是，MATLAB 语言总是以矩阵为单位进行计算的，因此必须掌握数组的创建、访问方法。读者借助于 MATLAB 本身所提供的各种数学函数和各种操作符可以编出很多实用程序。读者只要认真领会本章的内容并加强上机操作，就一定能够在很短的时间内达到基本了解 MATLAB 语言的目的。

# 第3章　MATLAB程序设计

**知识点：**

- 常用数学函数简介
- m文件与m函数
- 程序设计初步
- 文件操作的有关函数

MATLAB 提供了大量的常用数学函数命令，借助于 MATLAB 提供的各种运算符和丰富多彩的常用数学函数命令，就有可能编写出功能强大的应用程序。m 文件和 m 函数是两个重要的概念，需要认真领会。本章还将介绍 MATLAB 程序设计的基础知识以及 MATLAB 中有关文件操作的一系列函数。通过本章的学习，可以达到基本掌握MATLAB语言的目的。

## 3.1　常用的数学函数

MATLAB 的数学计算能力是其他软件所无法比拟的。本节简要介绍常用的数学函数在 MATLAB 中相应的实现命令，如表3-1所示。

**表3-1　常用的数学函数命令**

| | | |
|---|---|---|
| 三角函数 | sin | 正弦函数 |
| | sinh | 双曲正弦函数 |
| | asin | 反正弦函数 |
| | asinh | 反双曲正弦函数 |
| | cos | 余弦函数 |
| | cosh | 双曲余弦函数 |
| | acos | 反余弦函数 |
| | acosh | 反双曲余弦函数 |
| | tan | 正切函数 |
| | tanh | 双曲正切函数 |
| | atan | 反正切函数 |
| | atanh | 反双曲正切函数 |
| | atan2 | 四象限反正切函数 |
| | sec | 正割函数 |
| | sech | 双曲正割函数 |
| | asec | 反正割函数 |
| | asech | 反双曲正割函数 |

（续）

| | | |
|---|---|---|
| 三角函数 | Cot | 余切函数 |
| | coth | 双曲余切函数 |
| | acot | 反余切函数 |
| | acoth | 反双曲余切函数 |
| | csc | 余割函数 |
| | csch | 双曲余割函数 |
| | acsc | 反余割函数 |
| | acsch | 反双曲余割函数 |
| 其他常用数学函数 | abs | 实数的绝对值以及复数的模 |
| | conj | 求复数的共轭 |
| | real | 求复数的实部 |
| | imag | 求复数的虚部 |
| | sqrt | 求平方根 |
| | exp | 求指数 |
| | log | 自然对数 |
| | log10 | 常用对数 |
| | log2 | 以 2 为底的对数 |
| | airy | Airy 函数 |
| | besselh | 第三类 Bessel 函数 |
| | besseli, besselk | 修正的 Bessel 函数 |
| | besselj, bessely | 第一类 Bessel 函数 |
| | beta,betainc,betaln | Beta 函数 |
| | ellipj | 椭圆雅可比（Jacobi）函数 |
| | ellipke | 第一、二类完全椭圆积分 |
| | erf,erfc,erfcx,erfinv | Error 函数 |
| | gamma,gammainv,gammaln | Gamma 函数 |
| | legendre | 勒让德（Legendre）函数 |
| | lcm | 最小公倍数 |
| | gcd | 最大公约数 |
| | expint | 指数积分 |
| | sign | 符号函数 |

## 3.2 m 文件与 m 函数

MATLAB 输入命令的常用方式有两种：一种是直接在 MATLAB 的命令窗口中逐条输入 MATLAB 命令；二是 m 文件工作方式。当命令行很简单时，使用逐条输入方式还是比较方便的。但当命令行很多时（比如说几十行乃至成百上千行命令），显然再使用这种方式输入 MATLAB 命令，就会显得杂乱无章，不易于把握程序的具体走向，并且给程序的修改和维护带来了很大的麻烦。这时，建议采用 MATLAB 命令的第二种输入形式 m 文件工作方式。

m 文件工作方式，指的是将要执行的命令全部写在一个文本文件中，这样既能使程序显得简洁明了，又便于对程序的修改与维护。m 文件直接采用 MATLAB 命令编写，就像在 MATLAB 的命令窗口直接输入命令一样，因此调试起来也十分方便，并且增强了程序的交互性。

m 文件与其他文本文件一样，可以在任何文本编辑器中进行编辑、存储、修改和读取。利用 m 文件还可以根据自己的需要编写一些函数，这些函数也可以像 MATLAB 提供的函数一样进行调用。从某种意义上说，这也是对 MATLAB 的二次开发。

m 文件有两种形式：一种是命令方式或称脚本方式；另一种就是函数文件形式。两种形式的文件扩展名均是.m。

### 3.2.1 m 文件

当遇到输入命令较多以及要重复输入命令的情况时，利用命令文件就显得很方便了。将所有要执行的命令按顺序放到一个扩展名为.m 的文本文件中，每次运行时只需在 MATLAB 的命令窗口输入 m 文件的文件名就可以了。需要注意的是，m 文件最好直接放在 MATLAB 的默认搜索路径下（X:\MATLAB6p1\work，其中 X 表示 MATLAB 所在的硬盘），这样就不用设置 m 文件的路径了，否则应当用路径操作指令 path 重新设置路径。另外，m 文件名不应该与 MATLAB 的内置函数名以及工具箱中的函数重名，以免发生执行错误命令的现象。MATLAB 对命令文件的执行等价于从命令窗口中顺序执行文件中的所有指令。命令文件可以访问 MATLAB 工作空间里的任何变量及数据。命令文件运行过程中产生的所有变量都等价于从 MATLAB 工作空间中创建这些变量。因此，任何其他命令文件和函数都可以自由地访问这些变量。这些变量一旦产生就一直保存在内存中，只有对它们重新赋值，它们的原有值才会变化。关机后，这里变量也就全部消失了。另外，在命令窗口中运行 clear 命令，也可以把这些变量从工作空间中删去。当然，在 MATLAB 的工作空间窗口中也可以用鼠标选择想要删除的变量，从而将这些变量从工作空间中删除。

接下来，编写一个名为 test.m 的命令文件，用来计算矩阵 A 和矩阵 B 的和，然后计算所得矩阵的逆矩阵。其中矩阵 A=[1 2 3 ;4 5 6;7 8 0]，矩阵 B=[2 3 0;5 8 7;10 3 6]，矩阵 A 与 B 的和放在 C 矩阵中，C 的逆矩阵放在 invC 变量中。

在 MATLAB 桌面执行“File/New/M-file”命令（如图 3-1 所示），或直接单击 MATLAB 桌面上快捷栏中的□按钮，将出现如图 3-2 所示的窗口。

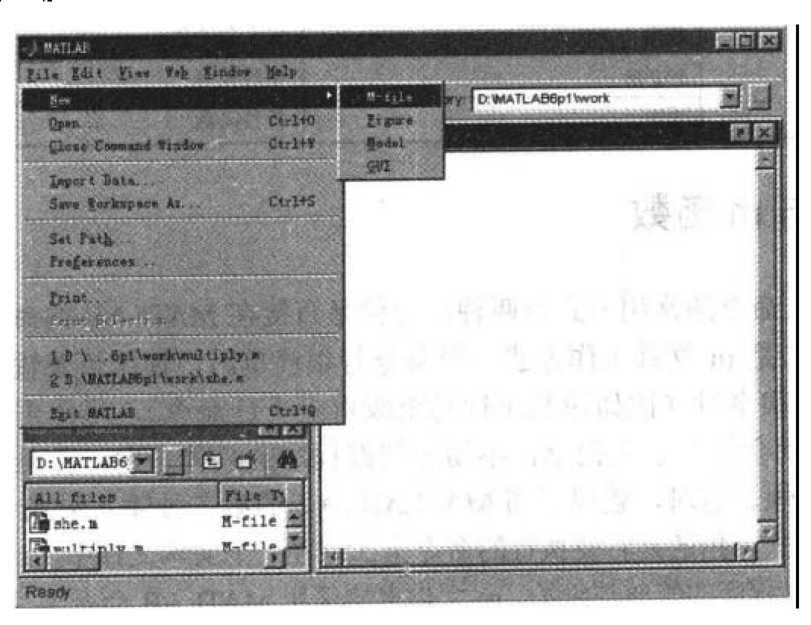



图 3-1 新建命令文件的选择次序

在如图 3-2 所示的窗口内依次输入命令语句：

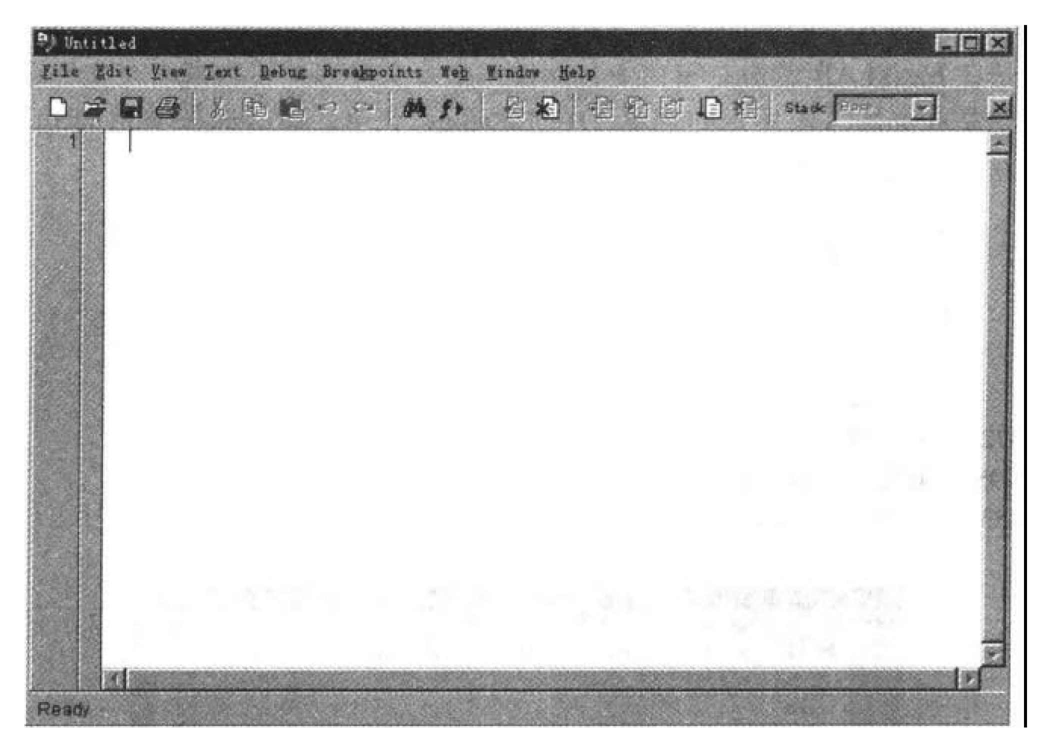



图 3-2　新建文件的操作窗口

```
A=[1 2 3 ;4 5 6;7 8 0];
B=[2 3 0;5 8 7;10 3 6];
C=A+B
invC=inv(C)
```

在输入过程中可以看到，在命令行的左侧自动显示了行号。输入完毕后，单击保存按钮 ，则弹出如图 3-3 所示的对话框。在对话框的“文件名”栏键入 test，然后单击保存按钮，就建立了名为 test.m 的命令文件，如图 3-4 所示。

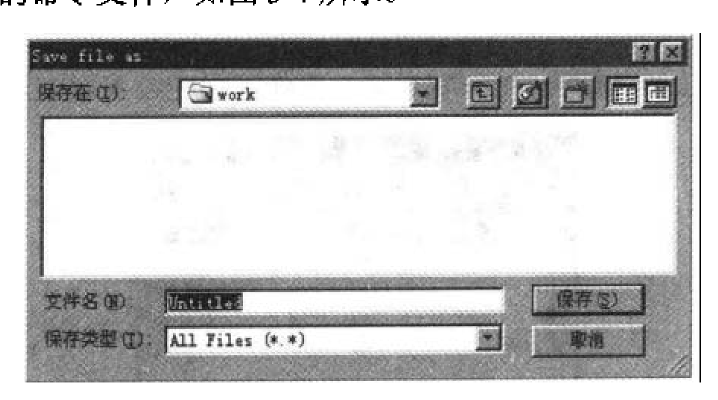



图 3-3　保存文件时出现的对话框

当然，在保存文件时也可以在主窗口界面中选择“File/Save”或者“File/Save As”，然后就会弹出图 3-3 所示的对话框，其他操作同上。

从以上过程可以看出，MATLAB 的操作界面已非常友好了，其菜单的分布风格与 Word 等软件的菜单分布格式非常相似。

通过上面的操作，已经建立了名为 test.m 的文件（从图 3-4 的标题栏上可以看出 test.m 文件保存在目录 d:\MATLAB6p1\work 下），现在就可以运行该文件了。运行命令文件有两种办法：一种是在 MATLAB 的命令窗口中直接键入命令文件名即可得到命令文件运行的结果；另外一种办法是在如图 3-4 所示的菜单栏中选 Debug 菜单，然后选择 Run 即可。随便选择一种运行办法，这时在 MATLAB 的命令窗口就会出现以下输出结果：

```
C =
     3     5     3
     9    13    13
    17    11     6
invC =
   -0.2372    0.0109    0.0949
    0.6095   -0.1204   -0.0438
   -0.4453    0.1898   -0.0219
```

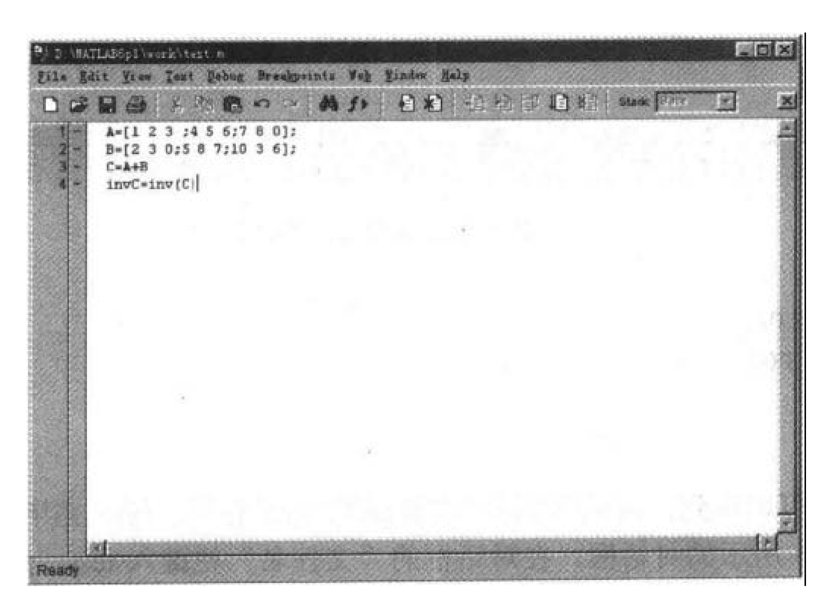



图 3-4　文件保存后的操作窗口

在 MATLAB 桌面上单击制表键，工作空间中会出现如图 3-5 所示的内容。

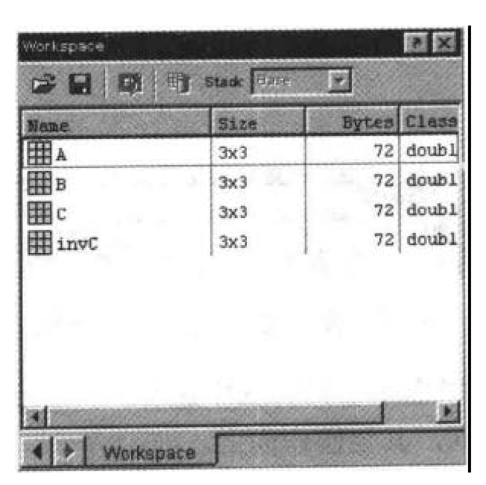



图 3-5　test 文件执行后出现的 Workspace 窗口

这说明 A、B、C 以及 invC 已经“到达”了 MATLAB 的工作空间。从工作空间的列表中还可以发现：只要是同一种类型的文件，那么它们在 MATLAB 的工作空间中是按照英文字母的顺序排列的，并且 MATLAB 默认的数值型数组是双精度的。

### 3.2.2 m 函数

m 函数文件是一个特殊的 m 文件，其常见格式如下：

function 返回变量列表=函数名（输入变量列表）

注释说明语句段

函数体语句

需要说明的是，这里输入变量的个数以及输出变量的个数是由 MATLAB 本身提供的两个保留变量 nargin 和 nargout 来给出的，它们分别是 Number of function input arguments 和 Number of function output arguments 的缩写形式。输入变量要用逗号隔开，输出变量多于 1 个时，要用方括号括起来。用户可以借助于 help 命令显示其中的注释说明语句段。通过这样的方法就可以建立函数文件或者称 m 函数，其调用方法与一般的 MATLAB 函数的调用方法相同。

函数文件相当于对 MATLAB 进行了二次开发。其作用与其他高级语言子函数的作用基本相同，都是为了实现特定目的而由用户自己编写的子函数。

函数文件与命令文件有着鲜明的区别：

- 函数文件的第一行必须包含 function 字符，命令文件无此要求。
- 函数文件的第一行必须指定函数名、输入参数及输出参数，命令文件无此要求。
- 一个函数文件可以有 0 个、1 个或多个输入参数和返回值。
- 函数文件要在文件的开头定义函数名，如 function [y1,y2]=func($x,a,b,c$)，则该函数文件名必须存为 func.m，而命令文件无此要求。
- 命令文件的变量在文件执行结束以后仍然保存在内存中而不会丢失，而函数文件的变量仅在函数运行期间有效（除非用 global 把变量说明成全局变量，否则函数文件中的变量均为局部变量），当函数运行完毕后，这些变量也就消失了。

需要说明的是，调用函数时所用的输入输出变量名并不要求与编写函数文件时所用的输入输出变量名相同。

下面讲述函数文件的创建以及函数的调用。

#### 1．函数文件的创建

下面以一个例子来说明函数文件的创建方法。

假如要对两个任意的变量 $a$、$b$ 进行处理，处理后要求分别得到 $a$ 的三次方，$b$ 的三分之一的结果值，则可以这样建立函数文件（假设函数文件名要求为 proab.m）：

与建立命令文件一样，在新建文件窗口中逐行输入以下命令：

```
function [y1,y2]=proab(a,b)
%Compute the result of a^3 and b/3
% a is the first input argument, b is the second input argument
y1=a^3;
y2=b/3;
```

然后把该文件保存为 proab（MATLAB 的默认文件后缀为.m），这样就建立了名为 proab 的函数。

**2．函数文件的调用**

读者可以像 MATLAB 中调用其他命令一样来调用自己创建的函数。例如，在 MATLAB 的命令窗口输入命令：

```
[x1,x2]=proab(2,3)
```

则得到的执行结果为：

```
x1 =
     8
x2 =
     1
```

这时就会在 MATLAB 的工作空间中出现：

| | | | |
|---|---|---|---|
| x1 | 1x1 | 8 | double array |
| x2 | 1x1 | 8 | double array |

由此可见，编写函数时所用的参数 $y1$、$y2$、$a$ 及 $b$ 都是局部变量，函数运行一结束，它们就消失了。

当然在命令窗口运行：

```
[x1 x2]=proab(2,3)
```

也会得到相同的结果。

如果在命令窗口输入命令：

```
help proab
```

按 Enter 键后就会出现以下信息：

```
 Compute the result of a^3 and b/3
  a is the first input argument, b is the second input argument
```

**3．函数的递归调用**

在调用函数的过程中出现直接或间接地调用该函数本身的现象称为函数的递归调用，并不是所有计算机语言都允许函数的递归调用。例如，FORTRAN 语言就不允许函数的递归调用而 C 语言允许。MATLAB 语言也允许函数的递归调用。如果函数 A 在执行过程中又直接调用了函数 A 本身，这就是函数的直接递归调用。若函数 A 在执行过程中没有调用函数 A 而调用了函数 B，但函数 B 在调用过程中又调用了函数 A，这就叫间接递归调用函数。在递归调用的函数中一般要有跳出递归调用的语句，否则函数会一直循环下去。

下面编写一个对输入的正整数求其阶乘的例子。函数名为 fun，要求能对输入的参数进行判断并给用户提供相应的信息。如果输入的是负整数，则跳出函数并告知用户输入的参数

不应当是负整数。

【例 3-1】

函数文件的清单如下：

```
function y=fun(n)
% Compute n!
if n<0
error('n should be positive integral or zero' ); %dispaly the error information return;
end
if n==0|n==1
    y=1;
else
    y=n*fun(n-1);
end
```

则在 MATLAB 的命令窗口中依次运行以下命令：

```
fun(3)
y2=fun(5)
y2=fun(-2)
```

得到的执行结果分别为：

```
ans =
      6
y2 =
    120
??? Error using ==> fun
n should be positive integral or zero
```

## 3.3 程序设计初步

本节主要讲述 MATLAB 程序设计方面的有关内容。

### 3.3.1 程序的结构

一般来说，程序的结构可以划分为 3 种：顺序结构、分支结构以及循环结构。从理论上讲，一门计算机语言只要有以上 3 种结构就可以编写出功能强大的程序了。由于 MATLAB 是由 C 语言编写的，因此在语法方面，它与 C 语言极为相似，但它比 C 语言要简单得多。

#### 1．顺序结构

顾名思义，顺序结构是指程序由依次按照顺序执行的各条语句组成。语句在程序文件中的位置就是程序的执行顺序。

以下程序就属于顺序结构的类型：

```
a=1;
```

```
b=2;
c=3;
d=a+b+c
```

当 MATLAB 执行以上程序时，逐行运行程序中的语句。

## 2．分支结构

分支结构又叫条件控制语句。分支结构在程序中占有重要的地位，因为在编程过程中总要对某些条件进行判断，从而根据判断的结果进行不同的后处理。MATLAB 提供了两种分支结构，分别是 if-else-end 和 switch-case-end。下面分别予以介绍。

（1）if-else-end 分支结构

这种分支结构又可以分成以下 3 种：

1）

```
if 表达式
  执行语句
end
```

☞ 提示：若表达式成立则执行 if-end 之间的执行语句，否则跳过该语句执行 end 后面的执行语句。

【例 3-2】

假设 $A$ 为 $m\times n$ 的矩阵，下面的分支结构可以用来判断 $A$ 矩阵的第 1 列元素是否全为 0，若全为 0 则将该列从 $A$ 中删除。

```
if A(:,1)==0
    A=A(1:m,2:n);
end
```

当然写成一行 if A(:,1)==0,A=A(1:m,2:n)，end 也是可以的，但这样书写，程序的结构看起来不是那么简明，因此建议分行书写。

2）

```
if 表达式
  语句 1
else
  语句 2
end
```

☞ 提示：如果表达式成立就执行语句 1，否则执行语句 2。

【例 3-3】

对于下面的语句：

```
if a<0
    disp('a is less than 0')
else
```

```
    disp('a is not les than 0')
end
```

如果 $a$=3，当执行到上述语句的 if 语句时，先判断 $a$ 是否小于 0，发现 $a$ 并不小于 0，因此执行 else 和 end 之间的语句：disp('a is not les than 0')。

3）

```
if 表达式 1
    语句 1
elseif 表达式 2
    语句 2
elseif  表达式 3
    语句 3
⋮     ⋮     ⋮
else
    语句 n
end
```

提示：先判断表达式 1，若成立则执行语句 1，执行完语句 1 后便终止执行该分支结构，即使后面有些表达式成立仍会被忽略掉，接着执行 end 以后的语句；若表达式 1 不成立则判断表达式 2，若表达式 2 成立则执行语句 2 且忽略后面的语句，执行完语句 2 以后，继续执行 end 之后的语句；若表达式 2 不成立则判断表达式 3…；若所有的表达式都不成立，则执行 else 和 end 之间的语句 n。

注意：该种分支结构中，else 语句可以不存在。

【例 3-4】

试编一函数用于计算分段函数：

$$y=\begin{cases}0 & x\leqslant 0\\ 1 & 0<x\leqslant 1\\ 2x & 1<x\leqslant 2\\ 2x+5 & x\geqslant 2\end{cases}$$

其程序如下：

```
function y=xy(x)   %创建名为 xy 的函数文件
if x<=0
    y=0;           %语句 1
elseif x<=1
    y=1;           %语句 2
elseif x<=2
    y=2*x;         %语句 3
else
    y=2*x+5;       %语句 4
end
```

将上述函数文件存为 xy.m 后，在 MATLAB 的命令窗口中输入：

y=xy(-2)

得到的执行结果为：

```
y =
     0
```

执行命令：

y=xy(2)

得到的执行结果为：

```
y =
     4
```

执行命令：

y=xy(6)

得到的执行结果为：

```
y =
       17
```

☞提示：在程序中若 $x$=-2 则执行语句 1，尽管语句 2、3、4 都成立也不会去执行它们。一般来说，分支结构中要用到大量的关系操作符（如>，<，>=，<=，==，~=）和逻辑操作符（如 isequal，isempty，ischar，isletter）。分支结构中的表达式既可以是标量间的计算，也可以是矩阵间的计算。

【例 3-5】
编制名为 test1.m 的命令文件，其清单如下：

```
A=[1 2 3
    4 5 6];
B=[1.5 2 3.4
    4.3 4.5 5.4];
if A<B
    disp('Matrix A<Matrix B');
elseif A<B+1
    disp('Matrix A<Matrix B+1');
else
    disp('not defined')
end
```

在 MATLAB 的命令窗口键入 test1 按 Enter 键后，显示结果为：

```
Matrix A<Matrix B+1
```

（2）switch-case-end 分支结构

此种分支结构能实现多分支功能。虽然 if-else-end 分支结构的第三种结构（形式 3）也可以实现多分支功能，但没有 switch-case-end 分支结构这么简明、易于维护。

switch-case-end 分支结构的表达形式为：

```
switch 表达式
case 常量表达式 1
        语句块 1
case 常量表达式 2
        语句块 2
 ⋮        ⋮       ⋮
case 常量表达式 n
        语句块 n
otherwise
        语句块 n+1
end
```

☞ 提示：switch 后面的表达式可以是任意类型，如字符串、矩阵、标量等；若表达式的值与 case 后面的某个常量表达式相等，则执行该 case 后的语句块。若表达式的值和所有的常量表达式的值都不相等，则执行 otherwise 后面的语句块。

【例 3-6】

已知收入和交税的关系为：

| 收入 | 税率 |
|---|---|
| 低于 5 万元 | 5 |
| 达到或超过 5 万元但低于 10 万元 | 10 |
| 达到或超过 10 万元但低于 15 万元 | 20 |
| 达到或超过 15 万元 | 35 |

则可以编制名为 tax 的函数文件，用它来求收入和交税的对应关系：

```
function y=tax(x)%     x 单位：元
a=fix(x/10000) ;       %将收入换算成万元并取整
switch a
case {0,1,2,3,4}
     y=x*0.05;
case {5,6,7,8,9}
     y=x*0.1;
case {10,11,12,13,14}
     y=x*0.2;
otherwise
     y=x*0.35;
end
```

在 MATLAB 命令窗口输入以下命令：

```
y=tax(120000)
```

得到的执行结果为：

```
y =
        24000
```

### 3. 循环结构

循环是计算机解决实际问题的主要手段。循环语句也涉及判断，只有满足一定的条件才执行循环，否则就会跳出循环。MATLAB 提供的循环有两种：for-end 循环和 while-end 循环。

（1）for-end 循环

其一般形式为：

```
for variable=expression
  statement
end
```

其中，variable 为循环变量名，expression 为循环变量表达式，statement 为语句块。在 expression 中给出循环变量的初值、步长和终值。通常用冒号来定义 expression，如 i:j:k 或者 i:j（前者初始值为 *i*、步长为 *j*、终止值为 *k*；后者初始值为 *i*、步长为 1、终止值为 *j*）。要注意的是，在 i:j:k 中若 $i<k$ 且 *j* 为负值，则是得不到矢量的。另外，variable 可以是字符串、字符串矩阵或者是由字符串组成的单元数组。

☞ 提示：在循环次数已知的情况下可以使用 for-end 循环结构。

循环是可以嵌套的。其一般表达形式为：

```
for variable1=expressionA
        statement1
        for variable2=expressionB
              statement2
        end
        statement3
end
```

【例 3-7】

假设矩阵 $A=\begin{bmatrix}5&1&0&0&0\\1&5&1&0&0\\0&1&5&1&0\\0&0&1&5&1\\0&0&0&1&5\end{bmatrix}$，下面用几种不同的方法创建矩阵 A。

方法 1：使用 for-end 循环创建矩阵 A（假设命令文件名为 mat1.m），程序清单如下：

```
tic                          %开始计时
A=[];                        %先将 A 赋为空阵
for i=1:5                    %对行进行操作
     for j=1:5               %对列进行操作
```

```
            if i==j
                A(i,i)=5;           %对角线赋值
            elseif abs(i-j)==1
                A(i,j)=1;           %行数、列数相差 1 的则赋值为 1
            else
                A(i,j)=0;           %其余的元素赋值为 0
            end
        end
end
A                                   %显示矩阵 A 的值
toc                                 %计时结束
```

在 MATLAB 的命令窗口键入：

```
mat1
```

按 Enter 键后，得到的执行结果为：

```
A =
     5     1     0     0     0
     1     5     1     0     0
     0     1     5     1     0
     0     0     1     5     1
     0     0     0     1     5
elapsed_time =
    0.1100
```

方法 2：仍然用 for-end 循环实现（假设命令文件名为 mat2.m）：

```
tic                         %开始计时
A=zeros(5);                 %先将 A 赋为零阵
for i=1:4
    A(i,i)=5;
    A(i,i+1)=1;
    A(i+1,i)=1;
end
A(5,5)=5;
A
toc                         %计时结束
```

在 MATLAB 的命令窗口先运行 clear 命令，然后运行 format long 命令。前者将工作空间清空，后者将数据显示格式改为长浮点型。然后键入 mat2 按 Enter 键后，显示结果为：

```
A =
     5     1     0     0     0
     1     5     1     0     0
     0     1     5     1     0
     0     0     1     5     1
     0     0     0     1     5
```

```
elapsed_time =
     0
```

方法 3：充分利用 MATLAB 提供的 diag 命令（假设命令文件名为 mat3.m）：

```
tic
A=[];
A=diag(5*ones(5,1))+diag(ones(4,1),1)+diag(ones(4,1),-1)
toc
```

在 MATLAB 的命令窗口先运行 clear 命令，然后运行 format long 命令。键入 mat3 按 Enter 键后，得到的执行结果为：

```
A =
     5     1     0     0     0
     1     5     1     0     0
     0     1     5     1     0
     0     0     1     5     1
     0     0     0     1     5
elapsed_time =
   0.05000000000000
```

由此可见，实现同一个问题可以用不同的方法。不同的人编写的程序不同，运行程序所耗时间也就不同。另外，在程序中把要使用的矩阵先赋值为空阵或者零阵，这是一个良好的编程习惯，这样处理可以改善内存的占用情况。

【例 3-8】

运行以下程序：

```
s='abc';              %循环变量可以是字符串
for i=s
    i
end
t={'aa' 'bb' 'cc'};   %循环变量可以是单元数组
for j=t
    j
end
```

得到的执行结果为：

```
i =
a
i =
b
i =
c
j =
    'aa'
j =
```

```
    'bb'
j =
    'cc'
```

while-end 循环

其表达形式为：

```
while 表达式
    循环体语句
end
```

当然，写成 while 表达式-循环体语句-end 也是可以的。

☞ 提示：表达式一般由逻辑运算和关系运算以及一般的运算组成，用于判断循环是否继续进行。一般来说，若循环的次数事先不好确定，这时用 while-end 循环就方便多了。while-end 循环中，只要表达式的值非 0 即为逻辑“真”，则程序就一直循环下去，直到表达式的值为 0 才停止循环。

【例 3-9】

已知 $\ln(1+x)$的 Maclaurin 展开式为：$\ln(1+x)=\sum_{k=1}^{\infty}\frac{(-1)^{k+1}x^k}{k}$，求 $x$=0.5 时由展开式得到的 ln(1+0.5)的值，迭代结束的条件是：要加入的下一项系数小于默认的精度 eps。要求程序给出所加项的项数。

程序清单为：

```
sum=0;              %置和的初值为 0
x=0.5;
k=1;      % k 为项数
while  abs((x^k)/k)>=eps
    sum=sum+(-1)^(k+1)*(x^k)/k;   %求和
    k=k+1;
end
k     %显示项数
sum  %显示求和的结果
```

运行结果为：

```
k =
    47
sum =
    0.4055
```

若在 MATLAB 窗口运行命令：

```
log(1.5)
```

则得到的执行结果为：

```
ans =
    0.4055
```

由此可见上述程序是正确的。

### 4. 循环的终止

在编制程序的过程中，有时会遇到终止循环的问题，这可以用命令 break 来实现。如果 break 命令位于嵌套循环的内循环，那么它只能终止内循环而外循环仍然继续进行。

【例 3-10】

通过迭代来求机器的最小正数是多少。这里提供两种实现方法：一种是在 for-end 循环中使用 break 语句；另一种则使用 while 循环。

方法 1：使用带 break 语句的 for 循环。

程序清单如下：

```
macheps=1;                %机器最小正数先置为 1
iter1=0;                  % iter1 为循环次数
for i=1:1000              %先把最大循环次数定为 1000
    macheps=macheps/2;
    iter1=iter1+1;
    if macheps+1<=1       %若 macheps 小于等于 0
        break             %跳出循环
    end
end
macheps=macheps*2         %显示最小机器正数
iter1                     %显示迭代次数
```

程序运行结果为：

```
macheps =
  2.2204e-016
iter1 =
    53
```

方法 2：使用 while 循环。

程序清单为：

```
macheps=1;
iter2=0;
while macheps+1>1         %当 macheps 大于等于 0 时执行循环
    macheps=macheps/2;
    iter2=iter2+1;
end
macheps=macheps*2
iter2
```

运行结果为：

```
macheps =
  2.2204e-016
iter2 =
     53
```

### 3.3.2 程序的调试

MATLAB 为用户调试程序提供了一套完整的交互式操作系统。以前版本的 MATLAB，其程序调试功能大多通过键盘命令来完成，现在 MATLAB 的调试比 C、FORTRAN 等语言的调试要简单得多。即使不对程序进行调试处理，MATLAB 在运行到程序的错误之处也会在 MATLAB 的命令窗口中自动给出警告信息以及出错信息，并给出程序中导致警告和错误命令产生的行号和列号。

一般来说调试程序有以下几个步骤：

步骤一：在程序的可疑处设置断点。

步骤二：执行程序。

步骤三：检查程序运行至断点处的各个变量的值。

步骤四：进行处理。

步骤五：恢复执行。

步骤六：结束调试状态。

步骤七：取消断点。

下面以一个简单的例子来简要说明程序的调试过程。

假如已经建立了一个名为 test.m 的命令文件，其程序清单如下：

```
x=-1:0.1:1;
y=-1:0.1:1;
z=x.*y;
plot(x,z);
```

在介绍该程序的调试以前，先来熟悉一下 MATLAB 提供的调试界面。

在建立上述命令文件的窗口（如图 3-6 所示），可以看到菜单栏中有 Debug 菜单以及 Breakpoints 菜单，这两个菜单是专门为调试程序提供的。其中 Debug 菜单的所含项如图 3-7 所示。Breakpoints 菜单的所含项如图 3-8 所示，其中 Stop If Error 选项、Stop If Warning 选项、Stop If NaN Or Inf 以及 Stop If All Error 选项分别代表当程序运行到错误处就停止、当程序运行到导致出现警告信息的语句时就停止、当程序运行过程中出现 NaN 或 Inf 时就停止以及若程序语句全错就停止。其中 NaN 代表非数值数（如 0.0/0.0），Inf 代表无穷大值（如 1.0/0.0）。上述四个选项用户可以全部选中（每次选中一项后，MATLAB 就自动在该项前打上对号）。Breakpoints 菜单中还有两个选择项：Set/Clear Breakpoint F12 以及 Clear All Breakpoints，前者可以用来设置/清除断点而后者则可以清除所有的断点。

在图 3-6 所示的窗口中，先将光标停在 test 命令文件的第一行，然后选择如图 3-8 中的 Set/Clear Breakpoint F12 选项。这时文件窗口会变成如图 3-9 所示的样子，其中第一行的圆点表示断点。

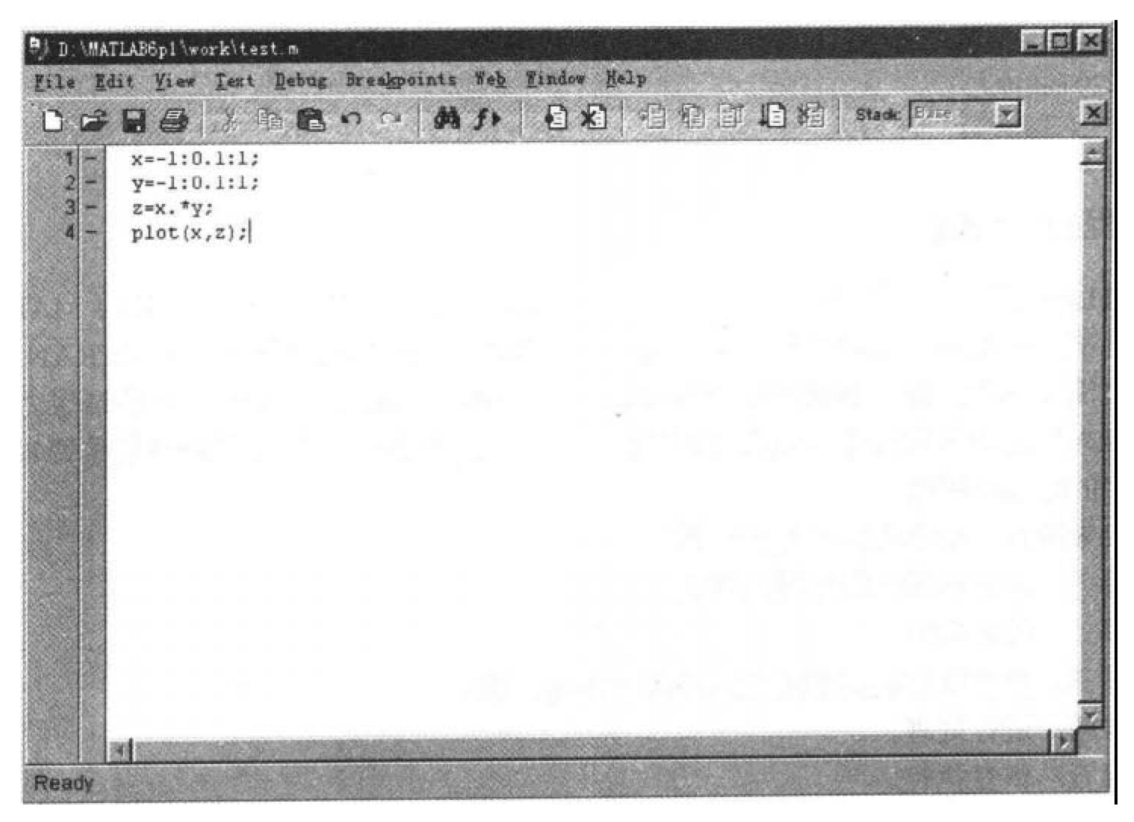



图 3-6　文件编辑窗口

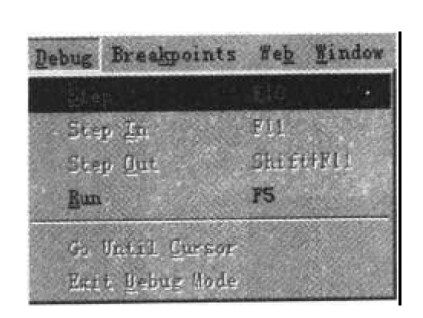



图 3-7　Debug 菜单栏

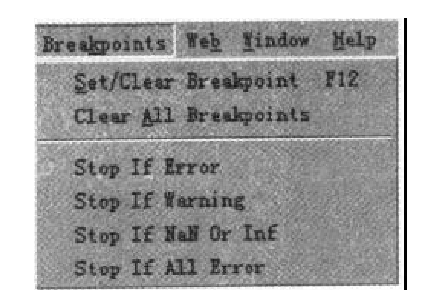



图 3-8　Breakpoints 菜单栏

这时在 MATLAB 的命令窗口执行 test 命令，命令窗口中会出现K>> 标志，表示程序正处于被调试状态。而文件窗口则变成如图 3-10 所示的图形，其中第一行处的箭头表示现在程序将执行第一行。

现在就可以选择 Debug 菜单中的 Step In F11 选项，该选项能逐步执行程序。当遇到 MATLAB 提供的函数时，则跟踪到该函数的内部。此时可发现每按一次 Step In F11，箭头就向下走一行。接连选择两次 Step In F11 选项后（或者连按两次 F11 键），这时在 MATLAB 的工作空间就会出现：

| | | | |
|---|---|---|---|
| x | 1x21 | 168 | double array |
| y | 1x21 | 168 | double array |

这从另一个角度说明 test 程序已经执行了两行。重复上面的过程，当箭头停在程序的第四行处，选择 Step In F11 后，则出现如图 3-11 所示的窗口。这表明程序已经跟踪进入绘图函数的内部，可以看出绘图函数 plot 所在的路径为 D:\MATLAB6p1\toolbox

\MATLAB\graphics。在图 3-11 中的 Debug 菜单中选择 Step Out [illegible] Shift+F11（因为 MATLAB 本身提供的函数应该没有问题），这时 MATLAB 就运行完了 test 程序，并且绘制出如图 3-12 所示的图形。相应地，此时 test 文件所在的编辑窗口变成如图 3-13 所示的图形，其中箭头的方向朝下，表示已经运行到程序的最后（即程序已经运行完毕）。

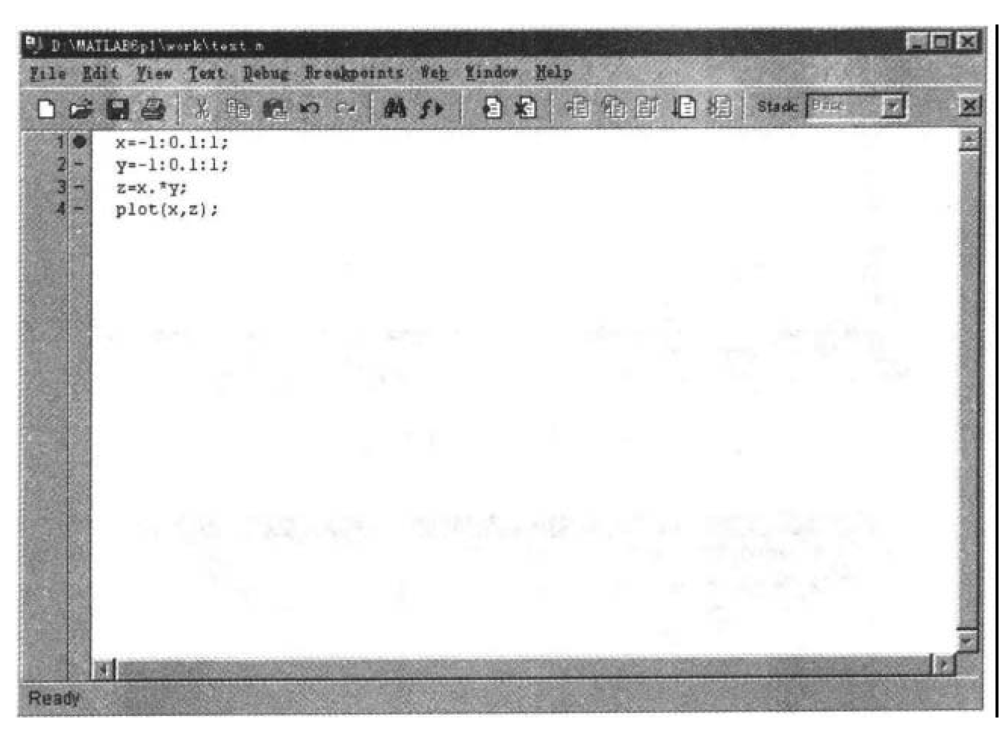



图 3-9　在程序的第一行处设置断点

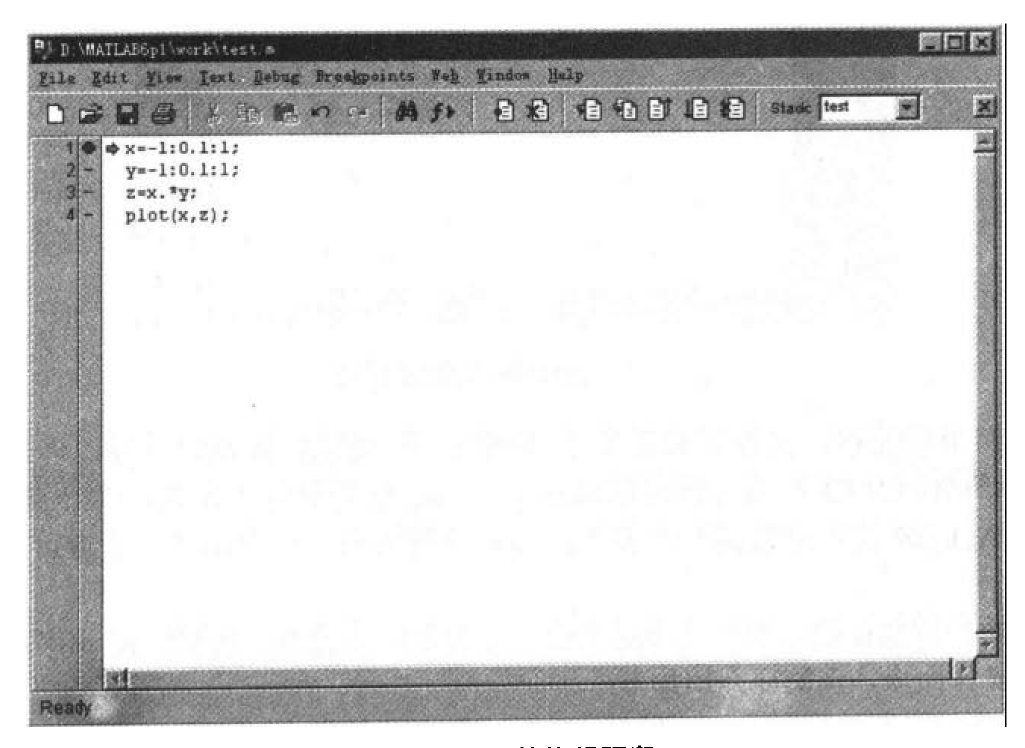



图 3-10　开始执行程序

```
D:\MATLAB6P1\toolbox\matlab\graphics\newplot.m
File Edit View Text Debug Breakpoints Web Window Help
27   %   is 'replace', or clear the current axes using CLA if its NextPlot
28   %   is 'replacechildren', or reuse the current axes as-is if its
29   %   NextPlot is 'add', or if no axes exist, create an axes.
30   %
31   %   See also HOLD, ISHOLD, FIGURE, AXES, CLA, CLF.
32
33   %   Copyright 1984-2001 The MathWorks, Inc.
34   %   $Revision: 5.12 $  $Date: 2001/04/15 11:59:51 $
35   %   Built-in function.
36
37 - if nargin == 0
38 -   hsave = [];
39 - elseif length(hsave) ~= 1 | ~ishandle(hsave)
40 -   error('bad handle')
41 - end
42
43 - fig = [];
44 - ax = [];
45
46 - if ~isempty(hsave)
47 -   obj = hsave;
48 -   while ~isempty(obj)
test.m  newplot.m
Ready
```

图 3-11 跟踪到绘图函数的内部

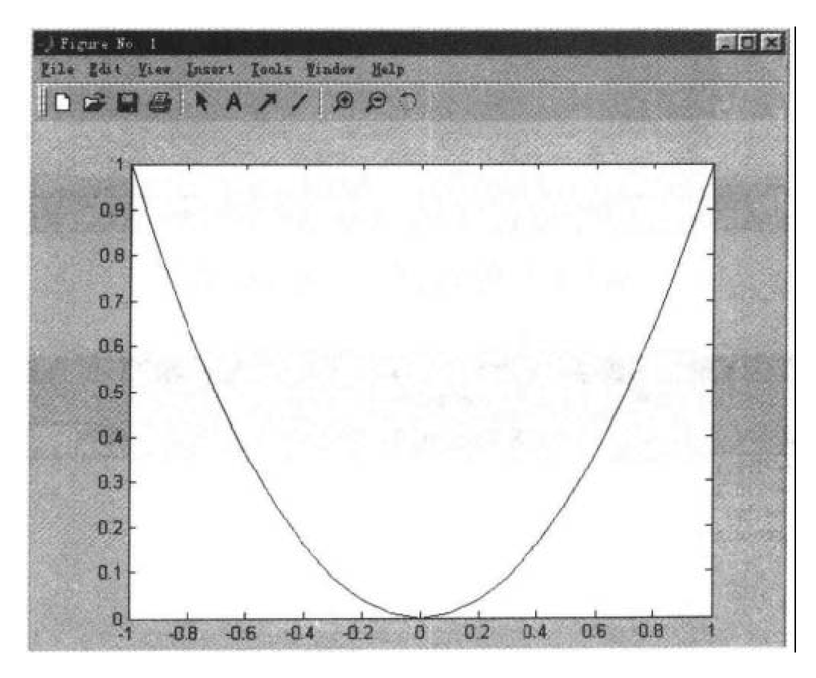



图 3-12 test 程序所绘制的图形

上面的例子很简单，只是简要说明了 MATLAB 调试器的使用过程。调试器 Debug 菜单中的 Go Until Cursor 选项可将程序由开始一直运行到光标所在处，Exit Debug Mode 选项可退出 MATLAB 的程序调试状态等等，这些功能就不一一介绍了，读者可以自己上机实践。

除了上面介绍的借助于交互式界面完成程序的调试功能外，MATLAB 还提供了一系列有关断点的操作命令，现简要介绍如下：

- dbstop in fname：在 fname.m 的第一可执行行处设置断点。
- dbstop an LineNo in fname：在 fname.m 的第 LineNo 行上设置断点。如果第 LineNo 行是不可执行的行，则程序会在运行 LineNo 行可执行的行后停止程序的运行。

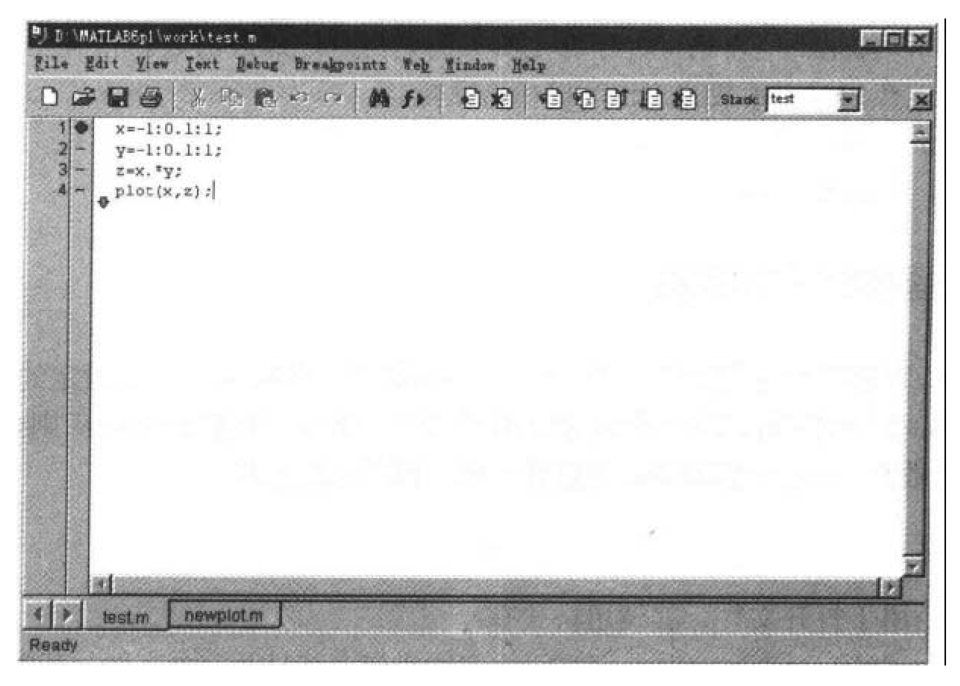



图 3-13 test 程序执行完毕后的窗口图形

- dbstop if v：当程序在发生由 v 所描述的错误时，停止运行并给出相应的提示信息，其中 v 可以是：

  error：当遇到错误时，停止程序的运行。

  NaN,Inf：当程序在运算过程中出现不定值或无穷大时，停止并退出程序的运行。

  warning：若有警告则停止并退出程序的运行。
- dbclear an LineNo in fname：清除文件 fname.m 第 LineNo 处的断点。
- dbclear all in fname：清除文件 fname.m 的所有断点。
- dbclear all：清除系统目录下的所有 m 文件中的断点（但不包括设为 erors 和 warning 的断点）。
- dbclear in fname：清除文件 fname.m 的第一可执行程序上的所有断点。
- dbclear if v：清除由 dbstop if v 设置的断点。
- dbstatus fname：在文件 fname 中列出所有的断点。
- mdbstatus：显示存放在 mdbstatus 中，用分号隔开的行数信息。

另外，了解以下命令对调试程序也是很有帮助的。

控制程序运行的命令，主要有以下几种：

- dbstep：运行 m 文件的下一行程序。
- dbstep nlines：执行下 nlines 行程序，然后停止。
- dbstep in：在下一个调用函数的第一可执行程序处停止运行。
- dbstep out：运行剩余的函数，当离开函数后则停止程序的运行。
- dbcont：执行所有行程序直至遇到下一个断点或者到达文件尾。
- dbquit：退出调试模式。

函数调用另外一个函数可以认为是函数的嵌套调用。MATLAB 使用栈来跟踪工作区和函数中的变量，下面的命令可用于在嵌套函数的工作区中进行切换：

- dbstep in：如果下一可执行程序行是函数的调用则跟踪该函数。
- dbup：切换到调用函数的工作区以检查变量。
- dbdown：切换回被调用函数的工作区。
- dbstack：显示嵌套函数调用的栈。

## 3.4 文件操作的有关函数

MATLAB 可以把产生的数据存入一个指定的文件，也可以从一个指定的文件中读取数据。MATLAB 在访问文件之前必须先把该文件打开，对文件操作完毕以后则要关闭该文件。以下列出的函数是专门为 MATLAB 文件进行读写操作而提供的。

### 1. fopen

fopen 命令用于打开文件，其调用格式有：

- fid=fopen('filename', 'permission')：以给定的操作方式打开一个文件。其中 filename 为将要打开的文件名，permission 为指定的操作方式。常用的操作方式有：'r'（只读）、'w'（只写）。对应后一种情况，当要打开的文件不存在时系统会自动建立一个新文件。'a' 则表示对文件进行追加，若要追加的文件不存在则创建该文件。要注意的是，这些操作控制符和'rb'、'wb'、'ab'等一样，都是针对二进制文件进行操作的。如果对文本格式的文件进行上述操作，则要在控制符后面加上字母 t，如'rt'、'wt'、'at'。其中 fid 为文件标识符，如果打开成功则返回一个正整数，如果失败则返回-1。另外有两个标准文件不用打开就可以直接操作，它们是：标准输出文件，fid=1；标准错误信息文件，fid=2。
- [fid,message]=fopen('filename', 'permission')：同上，但如果文件打开失败则返回信息 message。
- fopen('all')：列出用户当前已打开的全部文件标识符（标准输出文件和标准错误信息文件除外）。
- [filename, permission]=fopen(fid)：返回标识符为 fid 的文件名及操作方式。
- fopen('filename')：以'r'方式打开文件。

### 2. fclose

fclose 命令用于关闭文件，其调用格式有：

- fclose(fid)：关闭标识符为 fid 的文件。若关闭成功则返回 0，否则返回-1。
- fclose('all')：关闭标识符为 0、1、2 之外的所有已打开的文件。

### 3. fprintf

fprintf 命令用于文件的格式化输出，其调用格式为：

- count=fprintf(fid, 'format',A,...)：将输出表中的矩阵 A,...按格式控制串 format 的要求输出到标识符为 fid 的文件中。其中字符串 format 中含有格式说明，指出数据的类型和格式。例如，可以是%d（将输出表中的相应表达式值按整数输出），%10.4f（将输出表中的相应表达式值按浮点数输出，共占 10 位，其中含 4 位小数）。输出参数 count

为可选项，是 fprintf 输出的字节数。

### 4. fscanf1

fscanf 命令用于文件的格式化输入，其调用格式为：

- [A,count]=fscanf(fid, 'format',size)：按照控制字符串 format 的要求，从标识符为 fid 的文件中将数据输入 A 中。其中 count 是可选项，为 fscanf 的返回值（即成功返回的元素数）。size 也是可选项，给定读入元素的数目，默认值为整个文件。size 为整数 N，则读取 N 个元素；size 为 inf，则读到文件的末尾；size 为[M,N]，则表示读入一个 M×N 的矩阵。

【例 3-11】

下面的程序先计算 x,y 的值，然后将结果存入 fsize.dat。其程序清单如下：

```
x=0:0.1:1;                              %定义矢量
y=[x;sin(x)];                           %计算 y
fid=fopen('fsin.dat','wt')              %打开 fsin.dat 文件,对其进行写操作
fprintf(fid,'%5.2f   %10.5f\n',y);      %向文件 fsin.dat 写数据
fclose(fid);                            %关闭文件
type fsin.dat                           %在 MATLAB 的命令窗口中打印出 fsin.dat 文件
```

程序运行结果如下：

```
fid =
     3
 0.00      0.00000
 0.10      0.09983
 0.20      0.19867
 0.30      0.29552
 0.40      0.38942
 0.50      0.47943
 0.60      0.56464
 0.70      0.64422
 0.80      0.71736
 0.90      0.78333
 1.00      0.84147
```

这时可以在 Windows 的资源管理器中发现 fsin.dat 位于 D:\MATLAB6p1\work 路径下。当然也可以在进行写文件的操作时加入路径，如 fid=fopen('d:\calculate\fsin.dat','wt')，则程序运行后，fsin.dat 文件位于 d:\calculate 路径下。要注意的是，此时需将执行语句 type fsin.dat 改为 type d:\caculate\fsin.dat。

用户也可以使用 MATLAB 桌面上 File 菜单栏中的 Import Data 选项查看数据文件 fsin.dat，如图 3-14 所示。

### 5. fread

fread 命令可用于读取二进制数据，其调用格式为：

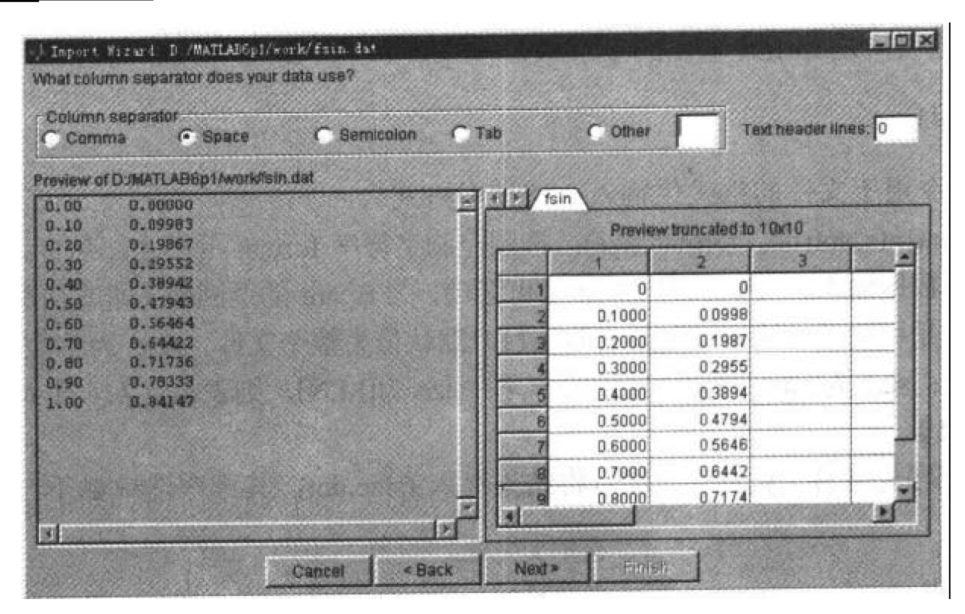



图 3-14 Import Wizard 对话框

● [A,count]=fread(fid,size)：从标识符为 fid 的文件中将二进制数据读入 A 中。size 是可选项，给定读入的个数，默认值为整个文件。size 为整数 N，则读取 N 个元素；size 为 inf，则读到文件的末尾；size 为[M,N]，则表示读入一个 M×N 的矩阵。

## 6. fwrite

fwrite 命令可以用于输出二进制数据，其调用格式为：

● count=fwrite(fid, A)：将数组 A 的元素写入标识符为 fid 的文件中，count 为写入成功的元素个数。

## 7. fgetl

fgetl 命令可以从文件中读入一行，其调用格式为：

● fgetl(fid)：从标识符为 fid 的文件中读入一行字符串，串中不包含后面的 Enter 换行符。当读到文件末尾时返回-1。

## 8. fgets

fgets 命令可以从文件中读入一行，其调用格式为：

● fgets(fid)：从标识符为 fid 的文件中读入一行字符串，串中包含后面的 Enter 换行符。当读到文件末尾时返回-1。

## 9. frewind

frewind 可将文件位置指针定位到文件首，其调用格式为：

● frewind(fid)：将标识符为 fid 的文件位置指针指向文件首。

## 10. fseek

fseek 命令可以对文件位置指针进行定位，其调用格式为：

- status=fseek(fid,offset,origin)：定位标识符为 fid 的文件的位置指针。其中 origin 为指定位置，offset 为偏移量。若 offset>0 则向后移，offset=0 则不动，offset<0 则向前移。origin=-1 为文件首，origin=0 为文件当前位置，origin=1 为文件尾。

### 11．ftell

ftell 命令可以得到文件位置指针的当前位置，其调用格式为：

- position=ftell(fid)：对标识符为 fid 的文件返回其位置指针的当前位置，若访问失败则返回-1。

## 3.5 小结

MATLAB 提供了大量的数学函数，借助于 MATLAB 本身所提供的各种数学函数和各种操作符就能实现很多实用程序。虽然 MATLAB 语言的程序结构比较简单，但是要想编制出简捷、省时、高效的程序就离不开仔细斟酌程序结构。m 文件与 m 函数是 MATLAB 中的两个重要概念，读者一定要认真辨析。本章还介绍了有关文件操作的指令，MATLAB 语言对文件的操作和 C 语言对文件的操作基本上是相同的，了解这一点对掌握 MATLAB 文件操作指令是大有裨益的。读者只要认真领会本章的内容并加强上机锻炼，就一定能够达到掌握 MATLAB 语言的目的。

# 第 4 章　符号运算概述

**知识点：**

- 创建符号变量、符号表达式、符号方程及符号矩阵
- 创建实数和复数
- 数值变量、符号变量及字符变量间的相互转换
- 基本操作命令
- 符号变量（表达式）的提取与代入

本章主要介绍 MATLAB 符号运算的基础知识。首先介绍符号变量的创建方法；其次介绍符号表达式、符号方程、符号矩阵、实数和复数的创建方法，以及数值变量、符号变量和字符变量之间的相互转换方法；最后介绍了 MATLAB 符号运算中常用的几个基本操作命令，并讲述了符号变量和符号表达式的提取与代入方法。通过对本章的学习，读者可以达到掌握 MATLAB 符号运算的基本知识。

在数学、物理及应用工程中除了数值计算外，还常常用到符号运算。一般的计算机语言（如 C 语言、FORTRAN 语言）实现数值运算还能轻松自如，但实现符号运算则相当困难。早期的 MATLAB 也不能实现符号运算，但随着 MATLAB 的开发以及为了适应用户的要求，MathsWorks 公司 1993 年购买了 MAPLE 的使用权。MAPLE 的长处就在于其强大的符号运算功能。MATLAB 拥有 MAPLE 后，开发出了实现符号运算的工具箱——Symbolic Math Toolbox。该工具箱可以通过以下三种方式实现符号运算：

- 用 MATLAB 语言编写的专用函数实现符号运算，包括：微积分、线性代数、化简、方程求解、变量的精度设定、积分变换、字符数值转换、基本操作、特殊函数、字符串处理功能……
- 利用 Maple.m 和 map.m 两个专门设计的 m 文件实现符号运算。使用这种符号运算方法需要掌握一定的 maple 基本语句。
- 利用 MATLAB 中的函数计算器 Function Calculator 进行简单的符号运算。该方法简便、直观。

本章和后面的各章将陆续讨论上述内容。

## 4.1　创建符号变量

MATLAB 中创建符号变量是利用命令 sym 和 syms 来实现。sym 命令用于创建单个符号变量，而 syms 命令则可以一次创建任意多个符号变量。因此，在符号运算中 syms 命令比 sym 命令常用。接下来，通过一些例子，说明如何创建符号变量。

【例 4-1】

执行以下程序：

```
A=sym('A')
beta=sym('beta')
x=sym('x')
```

执行结果为：

```
A =
A
beta =
beta
x =
x
```

这时可以用 size()命令来查看前面几个字符变量的大小：

```
SA=size(A)
sbeta=size(beta)
sx=size(x)
```

执行结果为：

```
SA =
     1     1
sbeta =
     1     1
sx =
     1     1
```

注意：符号型数据变量与字符型数据变量很容易混淆。其区别在于：字符型数据变量同数值型变量一样是以矩阵的形式进行保存的，而符号变量则不然。

【例 4-2】

执行以下程序：

```
beta1='beta'
y='a*b+c*d'
T='MATLAB Language 6.1'
N='(1+sqrt(2))/2'
```

执行结果为：

```
beta1 =
beta
y =
a*b+c*d
T =
MATLAB Language 6.1
N =
(1+sqrt(2))/2
```

此时用 size()命令来查看这四个字符变量的大小：

```
sbeta1=size(beta1)
sy=size(y)
sT=size(T)
sN=size(N)
```

执行结果为：

```
sbeta1 =
      1      4
sy =
      1      7
sT =
      1     19
sN =
      1     13
```

比较例 4-1 和例 4-2 的执行结果可知：符号型数据变量与字符型数据变量的存储形式是不一样的。

# 4.2 创建符号表达式及符号方程

## 4.2.1 创建符号表达式

在 MATLAB 中创建符号表达式有两种方法：一用 sym 命令直接创建符号表达式；二按普通书写形式创建符号表达式。总的来说，这两种方法各有利弊，但后一种方法比第一种方法常用一些。

### 1．用 sym 命令创建符号表达式

该种方法使用起来非常方便快捷。创建时不需在前面进行任何说明。但是需要注意的是：表达式内的符号变量并未得到说明，它们不会存在于 MATLAB 的工作空间。

【例 4-3】

执行名称为 he.m 的 MATLAB 程序：

```
clear                  %清空工作空间
f=sym('a*x+b')
f-a
```

执行结果为：

```
f =
a*x+b
??? Undefined function or variable 'a'.
```

这说明虽然创建了符号表达式 $a*x+b$ 并将它赋给了变量 $f$，但表达式中的各个符号变量

*a,x,b* 均未得到说明或创建，因而当执行 *f-a* 时，系统并不认识符号变量 *a*。

**2. 按普通书写形式创建符号表达式**

该种方法在创建符号表达式之前，必须把符号表达式所包含的全部符号变量都创建完毕。创建符号表达式时，只需按照赋值格式输入即可。

【例 4-4】

执行以下程序：

```
clear                    %清空工作空间
syms a x b
f=a*x+b
g=f-a
```

执行结果为：

```
f =
a*x+b
g =
a*x+b-a
```

### 4.2.2 创建符号方程

在 MATLAB 中，符号方程的创建方法与创建符号表达式的第一种方法类似。创建符号方程的通用格式为：

```
equ=('EQUATION')
```

【例 4-5】

执行以下程序：

```
equ1=('a*x^2+b*x+c=0')
e2=('a*x*y+z=b')
```

执行结果为：

```
equ1 =
a*x^2+b*x+c=0
e2 =
a*x*y+z=b
```

注意：创建符号方程时不能像创建符号表达式的第二种方法那样使用下面格式的命令：equ1=a*x^2+b*x+c=0。

## 4.3 创建符号矩阵

创建符号矩阵的方法比创建符号表达式、创建符号方程的方法要多一些。总的来说有以

下几种：用 sym 命令直接创建符号矩阵；用类似创建普通数值矩阵的方法创建符号矩阵；由数值矩阵转换为符号矩阵和以矩阵元素的通式来创建符号矩阵。接下来，将对不同的创建符号矩阵方法分别予以介绍。

### 4.3.1 用 sym 命令直接创建符号矩阵

用 sym 命令直接创建符号矩阵时，矩阵的元素可以是任何符号变量或符号表达式甚至是符号方程，并且元素的长度允许不等。输入符号矩阵时，矩阵行与行之间用“;”隔开，各矩阵元素之间用“,”或“空格”隔开。

【例 4-6】

执行以下程序：

```
mtrx=sym('[1/0 x y;x+y=0 tan(a/b) 2*c;a b*c 4^2]')
```

执行结果为：

```
mtrx =
[       1/0,        x,        y]
[     x+y=0,  tan(a/b),      2*c]
[         a,      b*c,      4^2]
```

由该例可以看出，用 sym 命令直接创建符号矩阵时，它对矩阵元素的要求很宽松。

### 4.3.2 用类似创建普通数值矩阵的方法创建符号矩阵

这种创建方法与按普通书写形式创建符号表达式的方法类似。同样要求在创建符号矩阵之前要将符号矩阵所包含的全部符号变量均创建完毕，然后在创建符号矩阵时只需按创建普通数值矩阵的格式输入即可。

【例 4-7】

执行以下程序：

```
syms x a b c
f=a*x+b;
g=a*x/c;
d=a+b+c;
equ1=sym('a*x^2+b*x+c=0');
M=[1 2 3 x
   f g d c
   equ1 5 b a]
```

执行结果为：

```
M =
[              1,        2,        3,        x]
[          a*x+b,    a*x/c,    a+b+c,        c]
[ a*x^2+b*x+c=0,        5,        b,        a]
```

### 4.3.3 由数值矩阵转化为符号矩阵

数值型变量和符号型变量是 MATLAB 中的两种不同变量类型。因此它们之间不能直接进行运算。为了实现数值型变量和符号型变量之间的各种运算，必须先将数值型变量转化为符号型变量。需要注意的是，这个转化过程是在系统内部自动完成的。

将数值矩阵 M 转化为符号矩阵 S 的命令为：

```
S=sym(M)
```

【例 4-8】

执行以下程序：

```
M=[1 2 3;4 5 6;7 8 9]
S=sym(M)
```

执行结果为：

```
M =
      1     2     3
      4     5     6
      7     8     9
S =
[ 1, 2, 3]
[ 4, 5, 6]
[ 7, 8, 9]
```

注意：矩阵 M 和矩阵 S 的显示形式是不一样的，读者很容易发现这一点。另外，无论矩阵 M 中的元素是以分数形式还是浮点数形式给出的，当 M 被转化为符号矩阵 S 以后，都将以最接近原来元素的精确有理形式给出，如例 4-9。

【例 4-9】

执行以下程序：

```
M=[2/3 0.333 0.3333;2^0.5 1.14 1.414;log(2) 1.087 1/0.3]
S=sym(M)
```

执行结果为：

```
M =
    0.6667    0.3330    0.3333
    1.4142    1.1400    1.4140
    0.6931    1.0870    3.3333
S =
[                    2/3,             333/1000,           3333/10000]
[                sqrt(2),                57/50,              707/500]
[ 6243314768165359*2^(-53),           1087/1000,                 10/3]
```

### 4.3.4 用矩阵元素的通式创建符号矩阵

当符号矩阵较大或者符号矩阵的元素较多时，如果仍是按部就班地逐个元素输入往往会事倍功半。这时应当仔细观察矩阵的各个元素间有无规律可循，如果矩阵的各个元素可以用含有行数、列数信息的通式表达出来，那么，毫无疑问，在这种情况下用矩阵元素的通式来创建符号矩阵是最为快捷的一种方法。

在早期版本的 MATLAB 中，只要在 sym 命令中加入几个相应的参数就可以创建用通式表达的符号矩阵。在新版本的 MATLAB 中 sym 命令没有提供这个功能，但可以通过自己编写函数的办法来实现用矩阵元素的通式来创建符号矩阵。

其代码如下：

```
function M=symat(row,column,frc)
%symat：函数的功能是利用矩阵元素的通式创建符号矩阵
%row：待创建符号矩阵的行数
%column：待创建符号矩阵的列数
%frc：矩阵元素的通式
for R=1:row
    for C=1:column
        c=sym(C);
        r=sym(R);
        M(R,C)=subs(sym(frc));
    end
end
```

【例 4-10】

试创建以下三个符号矩阵：

$$A=\begin{bmatrix} x+y & 2x+5y & 3x+9y & 4x+13y & 5x+17y \\ 4x+2y & 5x+6y & 6x+10y & 7x+14y & 8x+18y \\ 7x+3y & 8x+7y & 9x+11y & 10x+15y & 11x+19y \\ 10x+4y & 11x+8y & 12x+12y & 13x+16y & 14x+20y \\ 13x+5y & 14x+9y & 15x+13y & 16x+17y & 17x+21y \end{bmatrix}$$

$$B=\begin{bmatrix} \sin 1 & \sin 2 & \sin 3 \\ \sin 4 & \sin 5 & \sin 6 \\ \sin 7 & \sin 8 & \sin 9 \end{bmatrix}$$

$$C=\begin{bmatrix} e & e^5 & e^9 \\ e^2 & e^6 & e^{10} \\ e^3 & e^7 & e^{11} \\ e^4 & e^8 & e^{12} \end{bmatrix}$$

可以通过以下命令来实现：

```
syms x y c r                          %符号标量说明
a=(c+(r-1)*3)*x+(r+(c-1)*4)*y;        %矩阵 A 的元素通式
b=sin(c+(r-1)*3);                     %矩阵 B 的元素通式
d=exp(r+(c-1)*4);                     %矩阵 C 的元素通式
A=symat(5,5,a)                        %调用 symat 函数产生矩阵 A
B=symat(3,3,b)                        %调用 symat 函数产生矩阵 B
C=symat(4,3,d)                        %调用 symat 函数产生矩阵 C
```

执行结果为：

```
A =
[        x+y,     2*x+5*y,    3*x+9*y,    4*x+13*y,   5*x+17*y]
[    4*x+2*y,     5*x+6*y,   6*x+10*y,    7*x+14*y,   8*x+18*y]
[    7*x+3*y,     8*x+7*y,   9*x+11*y,   10*x+15*y, 11*x+19*y]
[   10*x+4*y,   11*x+8*y, 12*x+12*y,   13*x+16*y, 14*x+20*y]
[   13*x+5*y,   14*x+9*y, 15*x+13*y,   16*x+17*y, 17*x+21*y]
B =
[ sin(1), sin(2), sin(3)]
[ sin(4), sin(5), sin(6)]
[ sin(7), sin(8), sin(9)]
C =
[  exp(1),   exp(5),   exp(9)]
[  exp(2),   exp(6), exp(10)]
[  exp(3),   exp(7), exp(11)]
[  exp(4),   exp(8), exp(12)]
```

注意：由于在函数 symat 中使用了 M(R,C)=subs(sym(frc))的方法，因此当 f 为字符型参数时 symat 函数同样可以得出正确答案。读者可试运行以下命令：

```
A=symat(5,5,'(c+(r-1)*3)*x+(r+(c-1)*4)*y')
B=symat(3,3,'sin(c+(r-1)*3)')
C=symat(4,3,'exp(r+(c-1)*4)')
```

运行上述命令会得到与例 4-10 一样的结果。

## 4.4 创建实数和复数

前 3 节分别讲述了符号变量、符号表达式、符号方程以及符号矩阵的创建方法。本节将讨论 MATLAB 中符号变量实数和复数的创建方法。

在 MATLAB 中执行以下程序：

```
syms x y
a=real(x)
b=imag(x)
c=conj(x+y)
```

则执行结果为：

```
a =
1/2*x+1/2*conj(x)
b =
-1/2*i*(x-conj(x))
c =
conj(x+y)
```

由此可见，用 syms *x* 命令来创建符号变量，系统将默认所创建的变量 *x* 为复数。在实际应用当中，通常用到这样的表达式：$z=x+iy$（其中 $x,y\in R$）。这就提出了这样一个问题：在 MATLAB 中怎样表达一个符号型实数？sym 命令就实现了这个功能，其使用格式为：

```
x=sym(x,'real')
y=sym(y,'real')
```

或者：

```
syms x y real
z=x+y*i
```

通过以上命令创建的变量 *x,y* 就是实数型的。变量 *z* 是个虚数。这时用 real,iamg,conj 命令进行检查：

```
syms x y real
z=x+y*i
a=real(x)
b=imag(y)
c=conj(z)
```

显示结果为：

```
z =
x+i*y
a =
x
b =
0
c =
x-i*y
```

注意：若想清除变量 *x* 的实数属性，则只需使用以下命令：

```
x=sym(x,'unreal')
```

或者使用

```
syms x unreal
```

如果使用清除命令 clear x 将变量 *x* 从工作空间中清除，则当下次再创建符号变量 *x* 时，变量 *x* 仍将保持其实数性质；只有使用上述 sym 及 syms 命令才能清除变量的实数属性。

## 4.5 数值变量、符号变量及字符变量间的相互转换

在 MATLAB 中，数值变量、符号变量、字符变量的等级是不一样的。其中数值变量的级别最低；字符变量略高；符号变量级别最高。如果涉及到这三种变量的混合运算，则系统先将参与运算的所有变量自动统一转换为变量级别最高的类型，然后再进行计算。当然也可以通过 MATLAB 提供的命令来实现 3 种不同类型数据间的转换。按照转换目标的不同可以分为以下三类：将其他类型变量转换为符号变量；将其他类型变量转换为字符变量；将其他类型变量转换为数值变量。接下来，分别加以讨论。

### 4.5.1 将其他类型变量转换为符号变量

命令格式：

```
s=sym(f)
```

其中变量 $f$ 不受类型的限制，只要不是字符矩阵或非法的表达式，sym(f)命令均可将 $f$ 转换为符号变量 $s$。

【例 4-11】

执行以下程序：

```
s1=sym('1.234')
s2=sym(1.234)
f3='123f';
s3=sym(f3)
f4=['123';'456';'789'];
s4=sym(f4)
```

则显示结果为：

```
s1 =
1.234
s2 =
617/500
??? Error using ==> sym/sym (char2sym)
123f is not a valid symbolic expression.

Error in ==> D:\MATLAB6P1\toolbox\symbolic\@sym\sym.m
On line 92  ==>     S = char2sym(x);
```

可见 s3,s4 已给出出错信息，因为 f3,f4 分别是非法的表达式和字符矩阵。

### 4.5.2 将其他类型变量转换为字符变量

这里介绍两个命令：s=int2str(x)及 s=num2str(x)。它们分别能把整数和普通的数值变量转换为字符型变量。接下来，简要介绍一下它们的使用方法。

1. s=int2str(x)

该命令可以把整数 $x$ 转换为字符型变量 $s$。当 $x$ 为有理数时，将对 $x$ 先进行四舍五入得到整数后再把它转换为字符型变量。当 $x$ 为虚数时，将只对 $x$ 的实部进行转换，转换时按照将有理数转换为字符变量的相同规则进行。

注意：这里命令 int2str 即是 int to string（将整数转化为字符型变量、"2" 即是 "to"）

【例 4-12】
执行以下程序：

```
x1=-123;
s1=int2str(x1)
x2=-1.23;
s2=int2str(x2)
x3=-1.54;
s3=int2str(x3)
x4=-11.8+4*i;
s4=int2str(x4)
```

则显示结果为：

```
s1 =
-123
s2 =
-1
s3 =
-2
s4 =
-12
```

此时先用 clear 命令清除工作空间中的变量，运行上面的程序后，再用 whos 命令查看工作空间中的变量会发现：

```
Name      Size           Bytes  Class

s1        1x4                8  char array
s2        1x2                4  char array
s3        1x2                4  char array
s4        1x3                6  char array
x1        1x1                8  double array
x2        1x1                8  double array
x3        1x1                8  double array
x4        1x1               16  double array (complex)

Grand total is 15 elements using 62 bytes
```

显然 s1~s4 均变成了字符型变量。

### 2. s=num2str(x)

该命令可以把普通的数值型变量 x 转换为字符型变量 s，对 x 无任何限制。

【例 4-13】

执行以下程序：

```
clear
x1=-123;
s1=num2str(x1)
x2=-1.23;
s2=num2str(x2)
x3=-1.54;
s3=num2str(x3)
x4=-11.8+4*i;
s4=num2str(x4)
```

则显示结果为：

```
s1 =
-123
s2 =
-1.23
s3 =
-1.54
s4 =
-11.8+4i
```

再执行以下命令：

```
whos
```

则显示结果为：

```
  Name      Size           Bytes  Class
  s1        1x4                8  char array
  s2        1x5               10  char array
  s3        1x5               10  char array
  s4        1x8               16  char array
  x1        1x1                8  double array
  x2        1x1                8  double array
  x3        1x1                8  double array
  x4        1x1               16  double array (complex)

Grand total is 26 elements using 84 bytes
```

## 4.5.3 将其他类型变量转换为数值变量

这里主要介绍三种命令：double,str2num,numeric。

### 1. x=double(s)

当 $s$ 为符号变量时，x=double(s)命令将 $s$ 转换为数值变量 $x$。如果 $s$ 中含有非数字的符号则系统将给出出错信息；当 $s$ 为字符变量时，该命令将 $s$ 转换为数值矩阵 $x$。矩阵中元素的值为 $s$ 中相应字符的 ASCII 码值。

【例 4-14】

执行以下程序：

```
s1=sym(1.23);
x1=double(s1)
s2=sym('12*a');
x2=double(s2)
s3='12.345';
x3=double(s3)
s4='4/3';
x4=double(s4)
s5=['123';'1de';'1a2';'*>4'];
x5=double(s5)
```

则显示结果为：

```
x1 =
    1.2300
??? Undefined function or variable 'a'.

Error in ==> D:\MATLAB6P1\toolbox\symbolic\@sym\double.m
On line 45    ==> D = reshape(eval(X),m,n);
```

可见 x2 给出了出错信息。将 x2=double(s2)作为注释（在该命令行的开头处加上%即可），则执行结果为：

```
x1 =
    1.2300
x3 =
    49    50    46    51    52    53
x4 =
    52    47    51
x5 =
    49    50    51
    49   100   101
    49    97    50
    42    62    52
```

### 2. x=str2num(s)

x=str2num(s)命令专用于将字符变量 $s$ 转换为数值变量 $x$。当 $s$ 是一个包含非数字字符的

变量时，str2num(s)命令将返回一个空矩阵 *s*。

【例 4-15】

执行以下程序：

```
s1='1.23';
x1=str2num(s1)
s2='1/3';
x2=str2num(s2)
s3=['12';'34';'45';'67'];
x3=str2num(s3)
s4='12ab';
x4=str2num(s4)
```

执行结果为：

```
x1 =
    1.2300
x2 =
    0.3333
x3 =
    12
    34
    45
    67
x4 =
     []
```

### 3. x=numeric(s)

x=numeric(s)命令可将变量 *s* 转换为数值变量 *x*，这里 *s* 既可以是字符变量也可以是符号变量，但 *s* 不能是矩阵，否则将给出出错信息。

【例 4-16】

执行以下程序：

```
s1=sym('1/3');
x1=numeric(s1)
s2='1.234';
x2=numeric(s2)
s3='12aef';
x3=numeric(s3)
s4=['12';'34';'56';'78'];
x4=numeric(s4)
```

则执行结果为：

```
x1 =
    0.3333
```

```
x2 =
     1.2340
```

而 x3,x4 将给出以下出错信息：

```
??? Error using ==> sym/sym (char2sym)
12aef is not a valid symbolic expression.

Error in ==> D:\MATLAB6P1\toolbox\symbolic\@sym\sym.m
On line 92   ==>      S = char2sym(x);

Error in ==> D:\MATLAB6P1\toolbox\symbolic\numeric.m
On line 8   ==> N = double(sym(S));
```

## 4.6 基本操作命令

这里介绍符号运算中经常使用的几个操作命令：findsym、pretty 以及 MATLAB 提供的几种化简命令。其中 findsym 可以确定表达式中谁是自变量；而 pretty 以及 MATLAB 提供的其他化简命令则可以把符号运算的结果进行适当的化简。

### 4.6.1 findsym 命令

利用 findsym 命令可以找到符号表达式或者符号矩阵中的符号变量，其调用形式有以下两种：

- findsym(s)：其中 *s* 为一个标量或者是符号矩阵，执行该命令后返回一个字符串。该字符串包含 *s* 中出现的所有符号变量，并且符号变量是以字母顺序排列的（分别用“,”隔开）；如果 *s* 中没有找到符号变量则 findsym 将返回一个空的字符串。
- findsym(s,n)：返回字母表中最靠近字母“*x*”的 *n* 个字符变量。

【例 4-17】

执行以下程序：

```
syms alpha a b
findsym(alpha+a+b)
```

则显示结果为：

```
ans =
a, alpha, b
```

执行以下程序：

```
syms x1 y
findsym(cos(alpha)*b*x1+14*y,2)
```

则显示结果为：

```
ans =
x1,y
```

执行以下程序：

```
syms y
findsym(y*(4+3*i)+6*j)
```

则显示结果为：

```
ans =
y
```

执行以下程序：

```
syms x y z
findsym(x+i*y-j*z)
```

则显示结果为：

```
ans =
x, y, z
```

### 4.6.2 pretty 命令

在使用 MATLAB 的过程中，如果发现计算所得的结果显示的异常复杂时，可以使用 pretty 命令对计算的结果进行转化，从而使计算结果尽量符合表达习惯。

pretty 命令的调用格式为：

- pretty(A)：将代数式 $A$ 由机器格式转变为书写格式。需要注意的是，在转化过程中，MATLAB 不会对 $A$ 进行任何化简（包括展开）。

【例 4-18】

试用 pretty 命令将 $f=\dfrac{(x+y)\left(a+b^{c}\right)^{z}}{(x+a)^{2}}$ 和 $g=\dfrac{x\left(a+b^{c}\right)^{z}}{(x+a)^{2}}+\dfrac{y\left(a+b^{c}\right)^{z}}{(x+a)^{2}}$ 的 MATLAB 机器书写格式转化为手写格式。

执行以下程序：

```
syms x y z a b c
f=(x+y)*(a+b^c)^z/(x+a)^2
g=x*(a+b^c)^z/(x+a)^2+y*(a+b^c)^z/(x+a)^2
pretty(f)
pretty(g)
```

则显示结果为：

```
f =
(x+y)*(a+b^c)^z/(x+a)^2
g =
x*(a+b^c)^z/(x+a)^2+y*(a+b^c)^z/(x+a)^2
```

```
                    c   z
       (x + y) (a + b    )
       -------------------
                          2
               (x+ a)

             c   z              c   z
       x (a + b    )      y (a + b    )
       ------------ + ------------
                  2               2
         (x + a)          (x + a)
```

由执行结果可见，pretty 命令已经把符号表达式 $f$,$g$ 转化为很容易阅读的手写格式了。细心的读者会发现：事实上，例 4-18 中的 $f$ 和 $g$ 是完全相等的，但是 pretty 命令并没有解决判断 $f$ 和 $g$ 相等的问题。幸而，MATLAB 提供的化简功能正好解决了这个问题。

### 4.6.3 MATLAB 提供的化简命令

MATLAB 提供了多种化简符号表达式的方法，分别为：降幂排列法（collect）、展开法（expand）、重叠法（horner）、因式分解法（factor）、一般化简法（simplify）、不定化简法（simple）以及分子分母形式法（numden）。接下来，分别介绍以上方法。

1. 降幂排列法（collect）

collect 命令的调用格式有：

- collect(A)：按默认变量对表达式 $A$ 进行降幂排列，默认变量是指由 findsym 确定的变量。
- collect(A,v)：按照指定变量 $v$ 对表达式 $A$ 进行降幂排列。

【例 4-19】

（1）已知 $t=(ax^2+bx+c\text{-}3)^3\text{-}a(cx^5+4bx\text{-}1)\text{-}18b[(2+5x)^7\text{-}a+c]$，执行以下程序：

```
syms x   a b c
t=(a*x^2+b*x+c-3)^3-a*(c*x^5+4*b*x-1)-18*b*((2+5*x)^7-a+c);
tx=collect(t)                  %以默认变量对表达式 t 进行降幂排列
ta=collect(t,a)                %按变量 a 对表达式 t 进行降幂排列
```

则显示结果为：

```
tx =

-1406250*b*x^7+(a^3-3937500*b)*x^6+(3*b*a^2-a*c-4725000*b)*x^5+(-3150000*b+(c-3)*a^2+2*b^2*a+a*(2*(c-3)*a+b^2))*x^4+(-1260000*b+4*(c-3)*b*a+b*(2*(c-3)*a+b^2))*x^3+((c-3)*(2*(c-3)*a+b^2)+2*b^2*(c-3)+a*(c-3)^2-302400*b)*x^2+(3*(c-3)^2*b-4*b*a-40320*b)*x+(c-3)^3-18*b*(128+c-a)+a

ta =

x^6*a^3+3*(b*x+c-3)*x^4*a^2+(18*b-c*x^5-4*b*x+1+3*(b*x+c-3)^2*x^2)*a+(b*x+c-3)^3-18*b*((2+5*x)^7+c)
```

（2）已知表达式$f=x^2y+xy-x^2-2x$，$g=-\frac{1}{4}xe^{-2x}+\frac{3}{16}e^{-2x}$试将$f$按字变量$x$进行降幂排列，将$g$按$e^{-2x}$进行降幂排列。

执行以下程序：

```
syms x   a b c
f=x^2*y + y*x - x^2 - 2*x;
fx=collect(f)                    %按x对f进行降幂排列，与命令fx=collect(f,x)完全等效
g=-1/4*x*exp(-2*x)+3/16*exp(-2*x);
gepx=collect(g,exp(-2*x))            %按exp(-2x)对g进行降幂排列
```

则显示结果为：

```
fx =
(y-1)*x^2+(y-2)*x
gepx =
(-1/4*x+3/16)*exp(-2*x)
```

### 2. 展开法（expand）

在多项式、三角函数、对数函数和指数函数中经常用到该命令，其调用格式为：

- expand(S)：把符号表达式$S$展开。

【例 4-20】

执行以下程序：

```
syms x
expand((x+1)^3)
```

则显示结果为：

```
ans =
x^3+3*x^2+3*x+1
```

执行以下程序：

```
syms x y
expand(sin(x+y))
```

则显示结果为：

```
ans =
sin(x)*cos(y)+cos(x)*sin(y)
```

执行以下程序：

```
syms x y
v=[exp(x+y)   log(x^2/y)];
expand(v)
```

则显示结果为：

```
ans =
[exp(x)*exp(y), log(x^2)+log(1/y)]
```

### 3．重叠法（horner）

该化简方法是把符号表达式尽量化为 $ax(bx(cx\ldots(dx+z)+e+\ldots)+f)+g$ 的形式。其调用格式为：

- horner(S)：把符号表达式 $S$ 展成重叠式。

【例 4-21】

执行以下程序：

```
syms x
horner(x^3-6*x^2+11*x-6)
```

则显示结果为：

```
ans =
-6+(11+(-6+x)*x)*x
```

试用重叠法化简表达式：$x^5-4x^4+3x^3+5x^2+12$

执行以下程序：

```
syms x
horner(x^5-4*x^4+3*x^3+5*x^2+12)
```

显示结果为：

```
ans =
12+(5+(3+(-4+x)*x)*x)*x^2
```

### 4．因式分解法（factor）

因式分解法是 MATLAB 提供的诸多化简方法中最为常用的一种，其功能是把符号表达式进行因式分解化成连乘积的形式。其调用格式为：

- factor(S)：把符号表达式 $S$ 展成重叠式。

【例 4-22】

将下列式子进行因式分解：

（1）$f=x^3-3x^2-3x+1$

（2）$g=x^3-7x+6$

执行以下程序：

```
syms x
f=x^3-3*x^2-3*x+1;
g=x^3-7*x+6;
facf=factor(f)
facg=factor(g)
```

则显示结果为：

```
facf =
(x+1)*(x^2-4*x+1)
facg =
(x-1)*(x-2)*(x+3)
```

### 5．一般化简法（simplify）

MATLAB 提供的一般化简命令 simplify 充分考虑了符号表达式的各种运算法则，并充分考虑了各种特殊函数（如：三角函数、指数函数、对数函数、Bessel 函数、hypergeometric 函数、gamma 函数）的运算性质，经计算机比较后给出认为表达式相对简单的一种化简方法。用户无法知道所得的结果是由何种变换得到的。一般化简法在符号表达式的化简中有着广泛的应用。其调用格式为：

- simplify(S)：用一般化简法化简符号表达式 $S$。

【例 4-23】

用一般化简法化简下列各式：

（1）$a=x(x(x-2)+5)-1$

（2）$b=\log(xy)+\log z$

（3）$c=3xe^{x}e^{y+z}$

（4）$d$=besselj(2,$x$)-3besselj(1,$x$)

（5）$e=\sin^2 x+\cos^2 x-1$

执行以下程序：

```
syms x y z
a=x*(x*(x-2)+5)-1;
b=log(x*y)+log(z);
c=3*x*exp(x)*exp(y+z);
d=besselj(2,x)-3*besselj(1,x);
e=sin(x)^2+cos(x)^2-1;
sa=simplify(a)
sb=simplify(b)
sc=simplify(c)
sd=simplify(d)
se=simplify(e)
```

则显示结果为：

```
sa =
x^3-2*x^2+5*x-1
sb =
log(y*x)+log(z)
sc =
3*x*exp(x+y+z)
sd =
-(-2*besselj(1,x)+besselj(0,x)*x+3*besselj(1,x)*x)/x
se =
```

0

下面再回过头来看 4.6.2 节例 4-18 中提到的两个表达式：$f=\dfrac{(x+y)\left(a+b^c\right)^z}{(x+a)^2}$ 和 $g=\dfrac{x\left(a+b^c\right)^z}{(x+a)^2}+\dfrac{y\left(a+b^c\right)^z}{(x+a)^2}$。用一般化简法就可以判断 $f$ 与 $g$ 是否相等了，其执行程序如下：

```
syms x y z a b c
f=(x+y)*(a+b^c)^z/(x+a)^2;
g=x*(a+b^c)^z/(x+a)^2+y*(a+b^c)^z/(x+a)^2;
sf=simplify(f)              %用一般化简法对 f 进行化简,得到表达式 sf
sg=simplify(g)              %用一般化简法对 f 进行化简,得到表达式 sg
sf==sg                      %判断 sf 与 sg 是否相等
```

则显示结果为：

```
sf =
(x+y)*(a+b^c)^z/(x+a)^2
sg =
(x+y)*(a+b^c)^z/(x+a)^2
ans =
     1
```

逻辑计算结果为 1，说明表达式 $sf$ 与 $sg$ 相等从而 $f$ 与 $g$ 亦相等。

☞ 提示：MATLAB 提供的 6 种化简方法中，一般化简法（simplify）最为常用，但其目的性并不如其他几种方法明确。

### 6. 不定化简法（simple）

不定化简法是几种化简方法中最为笨拙的一种方法，但它综合了前面 5 种化简方法的优点。用不定化简法对表达式进行化简时，simple 命令自动将前面所讲的几种方法都尝试一遍，并且在化简过程中还考虑了多种转换方法（并不仅仅局限于 pretty 这一种转换方法），最后列出化简过程的所有结果。用户可以根据所列出的结果进行比较和筛选。不定化简法的调用格式为：

- simple(S)：用不定化简法化简符号表达式 $S$。若 $S$ 为矩阵，则返回结果为整个矩阵（而不是单个元素）的最简短表达形式。
- [R,HOW] = simple(S)：用不定化简法化简符号表达式 $S$，不列出所有结果而只给出表达式形式最短的一种化简结果。其中返回值 R 为 $S$ 的化简结果，HOW 为对应结果 R 所采用的化简方法或者转换方法。
- 不定化简法中经常用到的转化方法有：
  combine(trig)：以三角函数的运算性质对主对角代数式进行化简；
  convert(exp)：将代数式尽量转化为由 $e^x$、$e^{ix}$ 表示的指数形式；

convert(sincos)：将代数式尽量转化为由 sin(*x*)、cos(*x*)形式表示的式子；

congvert(tan)：将代数式尽量转化为由 tan(*x*)形式表示的式子。

【例 4-24】

（1）执行命令：

```
syms x
simple(sin(x)^2+cos(x)^2)
```

则显示结果为：

```
simplify:
1
radsimp:
sin(x)^2+cos(x)^2
combine(trig):
1
factor:
sin(x)^2+cos(x)^2
expand:
sin(x)^2+cos(x)^2
combine:
1
convert(exp):
-1/4*(exp(i*x)-1/exp(i*x))^2+(1/2*exp(i*x)+1/2/exp(i*x))^2
convert(sincos):
sin(x)^2+cos(x)^2
convert(tan):
4*tan(1/2*x)^2/(1+tan(1/2*x)^2)^2+(1-tan(1/2*x)^2)^2/(1+tan(1/2*x)^2)^2
collect(x):
sin(x)^2+cos(x)^2
ans =
1
```

（2）执行命令：

```
syms x
[R,HOW]=simple(sin(x)^2+cos(x)^2)
```

则显示结果为：

```
R =
1
HOW =
combine
```

（3）用[R,HOW] = simple(S)格式化简代数式的其他例子见表 4-1。

**表 4-1 不定化简法举例**

| S | R | How |
|---|---|---|
| cos(x)^2+sin(x)^2 | 1 | combine |
| 2*cos(x)^2-sin(x)^2 | 3*cos(x)^2-1 | simplify |
| cos(x)+(-sin(x)^2)^(1/2) | cos(x)+i*sin(x) | radsimp |
| cos(x)+i*sin(x) | exp(i*x) | convert(exp) |
| (x+1)*x*(x-1) | x^3-x | collect(x) |
| x^3+3*x^2+3*x+1 | (x+1)^3 | factor |
| cos(3*acos(x)) | 4*x^3-3*x | expand |

### 7. 分子分母法（numden）

numden 命令可以把符号变量表达式化简为有理形式，其中分子和分母是系数为整数、分子分母不含公约项的多项式。其调用格式为：

- [n,d]=numden(S)：用分子分母法化简符号表达式 *S*。返回结果 *n* 为分子，*d* 为分母。

【例 4-25】

（1）执行命令：

```
[n,d]=numden(sym(4/5))
```

结果显示为：

```
n =
4
d =
5
```

（2）执行命令：

```
syms x y
[n,d]=numden(x/y+y/x)
```

结果显示为：

```
n =
x^2+y^2
d =
y*x
```

由例 4-25 不难看出，当想把符号变量表达式表示为分子分母形式时，用 numden 命令是最简单的。

# 4.7 符号变量（表达式）的提取与代入

## 4.7.1 符号变量（表达式）的提取

解方程时经常遇到这种情况：所得的几个根中，有几个非常长的因子在解中出现多遍，使结果看起来或处理起来很不方便。MATLAB 提供的 subexpr 命令可以用一个语句完成筛选相同因子并整理表达式这样的复杂工作。

subexpr 命令的调用形式有以下几种：

- [Y,SIGMA]=subexpr(X,SIGMA)
- [Y,SIGMA]=subexpr(X,'SIGMA')
- Y=subexpr(X)

以上调用格式中：

- X：待整理的符号表达式或符号表达式的矩阵；
- SIGMA：在整理过程中提出的各种因子将以矩阵的格式保存在名为 SIGMA 的变量中；
- Y：已提取各种因子后，将整理完毕的符号表达式或者符号表达式的矩阵保存在 Y 中。

前两种调用格式功能完全一样，第三种调用格式和前两种调用格式所得的结果基本相同，只不过它将相同因子保存在默认名为 SIGMA 的变量中。

【例 4-26】

执行命令：

```
r=solve('a*x^3+b*x^2+c*x+d=0')
```

则显示结果为：

```
r =
[1/6/a*(36*c*b*a-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2-18*c*b*a*d+27*d^2*a^2+4*d*b^3)
^(1/2)*a)^(1/3)-2/3*(3*c*a-b^2)/a/(36*c*b*a-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2-18*c*b*
a*d+27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3)-1/3*b/a]
[ -1/12/a*(36*c*b*a-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2-18*c*b*a*d+27*d^2*a^2+4*d*b^3)^
(1/2)*a)^(1/3)+1/3*(3*c*a-b^2)/a/(36*c*b*a-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2-18*c*b*a*d+
27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3)-1/3*b/a+1/2*i*3^(1/2)*(1/6/a*(36*c*b*a-108*d*a^2-8*b^3+12*3^
(1/2)*(4*c^3*a-c^2*b^2-18*c*b*a*d+27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3)+2/3*(3*c*a-b^2)/a/(36*c*b*a-
108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2-18*c*b*a*d+27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3))]
[ -1/12/a*(36*c*b*a-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2-18*c*b*a*d+27*d^2*a^2+4*d*b^3)^
(1/2)*a)^(1/3)+1/3*(3*c*a-b^2)/a/(36*c*b*a-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2-18*c*b*a*d
+27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3)-1/3*b/a-1/2*i*3^(1/2)*(1/6/a*(36*c*b*a-108*d*a^2-8*b^3+12*3^
(1/2)*(4*c^3*a-c^2*b^2-18*c*b*a*d+27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3)+2/3*(3*c*a-b^2)/a/(36*c*b*a-
108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2-18*c*b*a*d+27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3))]
```

执行命令：

```
[root,sg]=subexpr(r,'sg')
```

则结果显示为：

```
root =
[ 1/6/a*sg^(1/3)-2/3*(3*c*a-b^2)/a/sg^(1/3)-1/3*b/a]
[ -1/12/a*sg^(1/3)+1/3*(3*c*a-b^2)/a/sg^(1/3)-1/3*b/a+1/2*i*3^(1/2)*(1/6/a*sg^(1/3)+2/3*(3*c*a -b^2)/a/sg
^(1/3))][ -1/12/a*sg^(1/3)+1/3*(3*c*a-b^2)/a/sg^(1/3)-1/3*b/a-1/2*i*3^(1/2)*(1/6/a*sg^(1/3)+2/3*(3*c*a-
b^2)/a/sg^(1/3))]
sg =
36*c*b*a-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2-18*c*b*a*d+27*d^2*a^2+4*d*b^3)^(1/2)*a
```

可见 subexpr 命令实现了筛选方程根中相同因子的功能。经过处理后的结果要简洁得多。如果把例 4-26 中的方程改为：$ax^3+bx^2+cx+d-5=0$

执行命令：

```
r=solve('a*x^3+b*x^2+c*x+d-5=0')
[root,sg]=subexpr(r,'sg')
```

则结果显示为：

```
r =
[ 1/6/a*(36*c*b*a+540*a^2-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2+90*c*b*a-18*c*b*a*d+675*
a^2-270*d*a^2-20*b^3+27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3)-2/3*(3*c*a-b^2)/a/(36*c*b*a+540*a^2-108*
d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2+90*c*b*a-18*c*b*a*d+675*a^2-270*d*a^2-20*b^3+27*
d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3)-1/3*b/a]
[ -1/12/a*(36*c*b*a+540*a^2-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2+90*c*b*a-18*c *b*a*d+
675*a^2-270*d*a^2-20*b^3+27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3)+1/3*(3*c*a-b^2)/a/(36*c*b*a+540*a^2-
108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2+90*c*b*a-18*c*b*a*d+675*a^2-270*d*a^2-20*b^3
+27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3)-1/3*b/a+1/2*i*3^(1/2)*(1/6/a*(36*c*b*a+540*a^2-108*d*a^2-
8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2+90*c*b*a-18*c*b*a*d+675*a^2-270*d*a^2-20*b^3+27*d^
2*a^2+4*d*b^3)^(1/2)*a)^(1/3)+2/3*(3*c*a-b^2)/a/(36*c*b*a+540*a^2-108*d*a^2-8*b^3+12*3^(1/2)
*(4*c^3*a-c^2*b^2+90*c*b*a-18*c*b*a*d+675*a^2-270*d*a^2-20*b^3+27*d^2*a^2+4*d*b^3)
^(1/2)*a)^(1/3))]
[ -1/12/a*(36*c*b*a+540*a^2-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2+90*c*b*a-18*c*b*a*d+
675*a^2-270*d*a^2-20*b^3+27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3)+1/3*(3*c*a-b^2)/a/(36*c*b*a+540*a^2-
108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2+90*c*b*a-18*c*b*a*d+675*a^2-270*d*a^2-20*b^3+
27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3)-1/3*b/a-1/2*i*3^(1/2)*(1/6/a*(36*c*b*a+540*a^2-108*d*a^2-8*b^3
+12*3^(1/2)*(4*c^3*a-c^2*b^2+90*c*b*a-18*c*b*a*d+675*a^2-270*d*a^2-20*b^3+27*d^2*a^2
+4*d*b^3)^(1/2)*a)^(1/3)+2/3*(3*c*a-b^2)/a/(36*c*b*a+540*a^2-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-
c^2*b^2+90*c*b*a-18*c*b*a*d+675*a^2-270*d*a^2-20*b^3+27*d^2*a^2+4*d*b^3)^(1/2)*a)^(1/3))]
root =
[ 1/6/a*sg^(1/3)-2/3*(3*c*a-b^2)/a/sg^(1/3)-1/3*b/a]
[ -1/12/a*sg^(1/3)+1/3*(3*c*a-b^2)/a/sg^(1/3)-1/3*b/a+1/2*i*3^(1/2)*(1/6/a*sg^(1/3)+2/3*(3*c*a-b^2)/a/sg^
(1/3))]
[ -1/12/a*sg^(1/3)+1/3*(3*c*a-b^2)/a/sg^(1/3)-1/3*b/a-1/2*i*3^(1/2)*(1/6/a*sg^(1/3)+2/3*(3*c*a-b^2)/a/sg
^(1/3))]
sg =
36*c*b*a+540*a^2-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2+90*c*b*a-18*c*b*a*d+675*a^2-
270*d*a^2-20*b^3+27*d^2*a^2+4*d*b^3)^(1/2)*a
```

可见“sg”也随之改变。

### 4.7.2 符号变量（表达式）的代入

使用 subs 命令可以方便地实现多种符号变量（表达式）的代入功能，其一般使用格式为：

- SS=subs(s,old,new)：s 为符号表达式，old 为 s 中将要被替代的“旧”变量名，new 是用来代替 s 中 old 的“新”变量名或代数式。
- SS=subs(s)：利用由调用函数或 MATLAB 工作空间中得到的具体值（无论是数值型还是字符型）来代替 s 中相应的所有变量。
- SS==subs(s,new)：用 new 来代替 s 中的自由符号变量。

注意：若执行命令 SS=subs(s,old,new)后发现返回值 SS 和符号表达式 s 相同，则系统将尝试执行 SS=subs(s,new,old)。之所以这样做是为了实现以前版本的 MATLAB 具有向后兼容以及避免记忆参数顺序而提供的一种附加功能。利用 SS=subs(s,old,new,0)可以避免上述操作。

【例 4-27】

（1）执行以下程序：

```
syms a b
s1=subs(a+b,a,4)            %把 a+b 中的 a 用 4 来代替
s2=subs(a+b,4,a)            %系统自动尝试执行 s2=subs(a+b,a,4)
s3=subs(a+b,4,a,0)          %关闭系统的自动尝试功能
```

则显示结果为：

```
s1 =
4+b
s2 =
4+b
s3 =
a+b
```

（2）执行以下程序：

```
a=980;C1=3;                 %对 a,C1 赋值
y=dsolve('Dy=-a*y')         %解微分方程 dy/dt=-ay
y=subs(y)                   %将所得结果 y 中的 a 和 C1 用指定的数值代入
```

则显示结果为：

```
y =
C1*exp(-a*t)
y =
3*exp(-980*(a*x^2+b*x+c-3)^3+980*a*(c*x^5+4*b*x-1)+17640*b*((2+5*x)^7-a+c))
```

（3）执行命令：

```
syms a b
subs(cos(a)+sin(b),{a,b},{sym('alpha'),2})
```

则显示结果为：

```
ans =
cos(alpha)+sin(2)
```

由此可见，subs 命令不但可以进行单一变量的替换而且可以同时进行多个变量的替换。

（4）执行命令：

```
syms a t
subs(exp(a*t),'a',-magic(2))
```

则显示结果为：

```
ans =
[     exp(-t), exp(-3*t)]
[ exp(-4*t), exp(-2*t)]
```

由此可见，subs 命令能对单个符号变量进行扩展。

（5）执行命令：

```
syms x y
subs(x*y,{x,y},{[0 1;-1 0],[1 -1;-2 1]})
```

则显示结果为：

```
ans =
     0    -1
     2     0
```

上述命令相当于执行了以下命令：

```
x=[0 1;-1 0];
y=[1 -1;-2 1];
x*y
```

（6）试将 $f=ax^2+bx+c$ 中的变量 $x$ 分别用 $y$ 和 $m+nt$ 代替。执行以下命令。

```
syms x y a b c              %符号变量说明
f=a*x^2+b*x+c;              %变量表达式赋值
sf1=subs(f,x,y)             %将变量表达式 f 中的 x 用 y 代替
sf2=subs(f,x,'m+n*t')       %将变量表达式中的 x 用 m+nt 代替
```

则显示结果为：

```
sf1 =
a*y^2+b*y+c
sf2 =
a*(m+n*t)^2+b*(m+n*t)+c
```

## 4.8 小结

本章主要介绍了 MATLAB 符号运算的基础知识。读者应当熟练掌握符号变量、符号表达式、符号方程、符号矩阵以及实数和复数的创建方法，适当了解数值变量、符号变量和字符变量之间的相互转换方法。另外，本章所介绍的基本操作命令、符号变量和符号表达式的提取与代入方法都是学习 MATLAB 语言所必备的知识，读者应当熟练掌握，并应结合上机实践本章所给出的各个例题。通过本章的学习，可以达到掌握 MATLAB 符号运算基本知识的目的。

# 第 5 章　常用符号运算功能的实现

**知识点：**

- 符号微积分与极限
- 级数
- 符号矩阵的有关操作命令
- 符号方程（方程组）的求解
- 反函数和复合函数的求法

本章所介绍的内容是本书的重点，主要介绍了 MATLAB 符号运算的具体实现方法。通过对本章的学习，读者将会发现 MATLAB 语言具备强大的符号运算功能：可以用 MATLAB 来求解微积分、极限，研究级数、符号方程、符号方程组、反函数和复合函数。总之，MATLAB 可以方便地解决常见的数学问题。

## 5.1　符号微积分与极限

### 5.1.1　符号微积分

当学习了第 2~4 章的有关 MATLAB 的基础知识以后，就能借助于 MATLAB 提供的符号运算命令进行一系列的符号运算了。

**1．符号求微分（diff）**

MATLAB 提供的 diff 命令可以求解导数和插分，其调用格式有：

- diff(X)：若 $X$ 为一矢量则 diff(X)的结果为插分，即[$X(2)$-$X(1)$,$X(3)$-$X(2)$,…,$X(n)$-$X(n$-1)]；若 $X$ 为一矩阵则 diff(X)的结果是对矩阵 $X$ 各列求插分所得到的矩阵，即[$X(2{:}n,:)$-$X(1{:}n$-1,:)]；若 $X$ 为 $N$ 维的数组则 diff(X)将按第一个非单元素集合的维进行插分计算。
- diff(X,N)：按第一个非单元素集合的维计算 $X$ 的 $N$ 阶导数。
- diff(X,N,DIM)：按维数 DIM 计算 $X$ 的 $N$ 阶导数。若 $N \geqslant$ size(X,DIM)则返回一个空矩阵。

☞ 提示：若指定对 $X$ 中的某个变量（如 $v$）进行求导数则可以用 diff(X,'v')实现，若想对 $X$ 中的某个变量（如 $v$）求 $n$ 阶导数则可以用 diff(X,'v',n)实现。

【例 5-1】

（1）分别计算 $f(x)=ax^2+bx+c$ 和 $g(x)=\sqrt{e^x+x\sin x}$ 的导数。可以执行命令：

```
syms x a b c                    %符号变量说明
f=a*x^2+b*x+c;                  %变量表达式赋值
g=(exp(x)+x*sin(x))^(1/2)       %变量表达式赋值
df=diff(f)                      %求变量表达式 f 关于 x 的导数
dg=diff(g)                      %求变量表达式 g 关于 x 的导数（复合函数求导数）
```

以上命令的执行结果为：

```
g =
(exp(x)+x*sin(x))^(1/2)
df =
2*a*x+b
dg =
1/2/(exp(x)+x*sin(x))^(1/2)*(exp(x)+sin(x)+x*cos(x))
```

*g* 的表达式较为复杂，可以用 pretty(dg)来转换一下。可以执行命令：

```
pretty(dg)
```

以上命令的执行结果为：

```
                 exp(x) + sin(x) + x cos(x)
         1/2 ------------------------------
                                        1/2
                   (exp(x) + x sin(x))
```

（2）计算矢量 $x$=（1 4 9 16 25 36 49）的插分。可以执行命令：

```
diff((1:7).^2)    %用 diff 命令求矢量的插分
```

以上命令的执行结果为：

```
ans =
     3     5     7     9    11    13
```

（3）已知矩阵 $X=\begin{bmatrix}3 & 7 & 5\\0 & 9 & 2\end{bmatrix}$，执行以下命令：

```
X=[3 7 5
   0 9 2];
dx11=diff(X,1,1)     %按照 X 的列计算插分
dx12=diff(X,1,2)     %相当于按照 X 的行计算插分
dx22=diff(X,2,2)     %相当于按照 X 的行计算 2 次插分
dx32=diff(X,3,2)     %由于 3=size(X,2)，故结果必定为一空矩阵
```

以上命令的执行结果为：

```
dx11 =
    -3     2    -3
dx12 =
```

```
        4     -2
        9     -7
dx22 =
      -6
-16
dx32 =
   Empty matrix: 2-by-0
```

（4）求函数 $f(x)=\log(x^3)$的导数。可以执行命令：

```
syms x
f=log(x^3);
g=diff(f)                %求对数函数的导数
```

以上命令的执行结果为：

```
g =
3/x
```

（5）求函数 $f(x)=e^{-2x}\cos\left(3\sqrt{x}\right)$ 的 3 阶导数。可以执行命令：

```
syms x
f=exp(-2*x)*cos(3*x^0.5)
g=diff(f,3)                 %求复合函数的 3 阶导数
```

以上命令的执行结果为：

```
f =
exp(-2*x)*cos(3*x^(1/2))
g =
-8*exp(-2*x)*cos(3*x^(1/2))-18*exp(-2*x)*sin(3*x^(1/2))/x^(1/2)+27/2*exp(-2*x)*cos(3*x^(1/2))/x-9/8*
exp(-2*x)*sin(3*x^(1/2))/x^(3/2)+27/8*exp(-2*x)*cos(3*x^(1/2))/x^2-9/8*exp(-2*x)*sin(3*x^(1/2)) /x^(5/2)
```

再执行命令：

```
pretty(g)                   %用 pretty 命令把 g 转化为简洁的表达式
```

则得到的命令执行结果为：

```
                                                  1/2
                         1/2          exp(-2 x) sin(3 x    )
   -8 exp(-2 x) cos(3 x     ) - 18 ----------------------
                                              1/2
                                             x


                                1/2                         1/2
              exp(-2 x) cos(3 x     )        exp(-2 x) sin(3 x     )
     + 27/2 ----------------------- - 9/8  ----------------------
                      x                               3/2
                                                     x
```

```
                     1/2                    1/2
      exp(-2 x) cos(3 x    )     exp(-2 x) sin(3 x    )
+ 27/8 ---------------------- - 9/8 ----------------------
               2                          5/2
              x                          x
```

（6）求导数 $\frac{\mathrm{d}}{\mathrm{d}y}\left[\frac{x\sin\left(e^{\sqrt{y}}\right)}{z}\right]$。可执行以下命令：

```
syms x y z
f=x*sin(exp(y^0.5))/z;             %变量表达式赋值
g=diff(f,y)                        %对指定变量 y 求导数
```

以上命令的执行结果为：

```
g =
1/2*x*cos(exp(y^(1/2)))/y^(1/2)*exp(y^(1/2))/z
```

再执行命令：

```
pretty(g)                          %把 g 转化为较为简洁的形式
```

则得到的命令执行结果为：

```
                 1/2        1/2
       x cos(exp(y    )) exp(y    )
  1/2 ---------------------------
                  1/2
                 y    z
```

（7）求下列导数 $\frac{\mathrm{d}}{\mathrm{d}x}\left[f(x,y,z)\right]$，$\frac{\mathrm{d}}{\mathrm{d}z}\left[f(x,y,z)\right]$，$\frac{\mathrm{d}^2}{\mathrm{d}y^2}\left[f(x,y,z)\right]$，$\frac{\mathrm{d}}{\mathrm{d}z}\left\{\frac{\mathrm{d}^2}{\mathrm{d}y^2}\left[f(x,y,z)\right]\right\}$ 和 $\frac{\mathrm{d}}{\mathrm{d}z}\left[\sin\left(a^x+y^b+c^z\right)\right]$。可以执行以下命令：

```
syms x y z a b c
f=sym('f(x,y,z)');                 %变量表达式赋值
g=sin(a^x+y^b+c^z) ;               %变量表达式赋值
pretty(diff(f))                    %对隐函数求导数，并转化为较为简洁的结果
pretty(diff(f,z))                  %对隐函数求导数（对指定变量 z 求导数），并把结果转
                                   %化为较为简洁的形式
pretty(diff(f,y,2))                %求 f 关于 y 的 2 阶导数
pretty(diff(diff(f,y,2),'z'))      %先求 f 关于 y 的 2 阶导数，再对所得的结果求关于 z
                                   %的导数
pretty(diff(g,z))                  %求 g 关于 z 的导数
```

则得到的命令执行显示结果为：

```
  d
 --- f(x, y, z)
  dx

  d
  -- f(x, y, z)
  dz

    2
   d
 ---- f(x, y, z)
    2
   dy

     3
    d
 ------- f(x, y, z)
      2
 dz dy

          x      b    z    z
 cos(a      + y   + c   ) c   log(c)
```

2．符号求积分（int）

MATLAB 提供的 int 命令既可以计算不定积分又可以计算定积分、广义积分，接下来，将其进行逐一介绍。

1）求不定积分

● int(S)

● int(S,v)

上述两种调用格式都是求 $S$ 的不定积分，区别在于：第 1 种调用格式以 findsym(S)命令寻找到的变量为自变量，计算 $S$ 的不定积分；而第 2 种调用格式则针对指定变量 $v$ 进行不定积分的计算。

注意：若 $S$ 为常数，那么 int(S)的计算结果为该常数和 $x$ 的乘积形式。

【例 5-2】

（1）分别计算函数 sin(*xy*+*z*)关于 *x*,*z* 的不定积分。可以执行以下命令：

```
syms x y z
f=sin(x*y+z);            %变量表达式赋值
fx=int(f)                %求 f 关于 x 的不定积分
fz=int(f,z)              %求 f 关于 z 的不定积分
```

以上命令的执行结果为：

```
fx =
-1/y*cos(y*x+z)
fz =
-cos(y*x+z)
```

（2）已知矩阵 $A=\begin{bmatrix}\cos(xt) & \sin(xt)\\ -\sin(xt) & \cos(xt)\end{bmatrix}$，求矩阵 $A$ 关于变量 $t$ 的积分，可以执行以下命令：

```
syms x t
A=[cos(x*t),sin(x*t);-sin(x*t),cos(x*t)];
int(A,t)             %求符号矩阵的不定积分
```

以上命令的执行结果为：

```
ans =
[ 1/x*sin(x*t),   -cos(x*t)/x]
[    cos(x*t)/x, 1/x*sin(x*t)]
```

（3）执行以下命令：

```
syms x alpha t
a=int(1/(1+x^2))                 %计算 1/（1+x^2）的不定积分
b=int(besselj(1,x),x)            %计算第一类 Bessel 函数的不定积分
c=int([exp(t) exp(alpha*t)])     %求矩阵的不定积分
d=int(sin(alpha*x),alpha)     %自变量设为 alpha
```

以上命令的执行结果为：

```
a =
atan(x)
b =
-besselj(0,x)
c =
[exp(t), 1/alpha*exp(alpha*t)]
d =
-cos(alpha*x)/x
```

2）求定积分与广义积分

- int(S,v,a,b)：其中 $S$ 为表达式，$v$ 为指定变量，$a$ 为积分下限，$b$ 为积分上限，若省略 $v$ 则针对系统的默认变量计算定积分。当 $a$ 或 $b$ 取 inf(或-inf)时，该命令计算的是广义积分。

【例 5-3】

（1）计算定积分：$\int_0^{\frac{\pi}{6}}(\sin x+2)\mathrm{d}x$、$\int_0^{\frac{\pi}{3}}x^y\mathrm{d}y$ 和 $\int_2^{\sin t}4tx\mathrm{d}x$。可以执行以下命令：

```
syms x y t
f=sin(x)+2;
```

```
fx=int(f,0,pi/6)
fz=int(x^y,y,0,pi/3)
fh=int(4*x*t,x,2,sin(t))
```

以上命令的执行结果为：

```
fx =
-1/2*3^(1/2)+1/3*pi+1
fz =
(x^(1/3*pi)-1)/log(x)
fh =
2*t*(sin(t)^2-4)
```

由此可知，直接由 int 命令计算的定积分结果是个表达式，若希望得到具体的数值则可借助于 numeric 命令。

☞ 提示：计算定积分时，int 命令经常和 numeric 命令连用。

再执行命令：

```
syms x
f=sin(x)+2;
fx=numeric(int(f,0,pi/6))
```

则得到的执行结果为：

```
fx =
    1.1812
```

（2）计算广义积分 $\int_1^{+\infty}\frac{1}{x^2}\mathrm{d}x$、$\int_1^{+\infty}\frac{1}{x^2+1}\mathrm{d}x$。可以执行以下命令：

```
syms x
f=1/x^2;
g=1/(1+x^2);
fx=int(f,1,inf)
fz=int(g,1,inf)
```

以上命令的显示结果为：

```
fx =
1
fz =
1/4*pi
```

（3）计算广义积分 $\int_{-\infty}^{+\infty}\frac{1}{4x^2+2x+3}\mathrm{d}x$、$\int_{-\infty}^{+\infty}\frac{1}{x^2+2x+1}\mathrm{d}x$ 和 $\int_{-\infty}^{+\infty}\frac{1}{x^2-2x+3}\mathrm{d}x$。可以执行以下命令：

```
syms x
f=1/(4*x^2+2*x+3);
```

```
g=1/(x^2+2*x+1);
h=1/(x^2-2*x+3);
fx=numeric(int(f,-inf,inf))
fg=int(g,-inf,inf)
fh=int(h,-inf,inf)
```

则得到的执行结果为：

```
fx =
    0.9472
fg =
inf
fh =
1/2*pi*2^(1/2)
```

由计算结果可见 $\int_{-\infty}^{+\infty}\frac{1}{x^2+2x+1}dx$ 并不存在。事实上，当 $x\in[-50,50]$时，$\frac{1}{x^2+2x+1}$ 的图像如图 5-1 所示，由图可知函数的图像并不连续，从而该函数的广义积分并不存在。

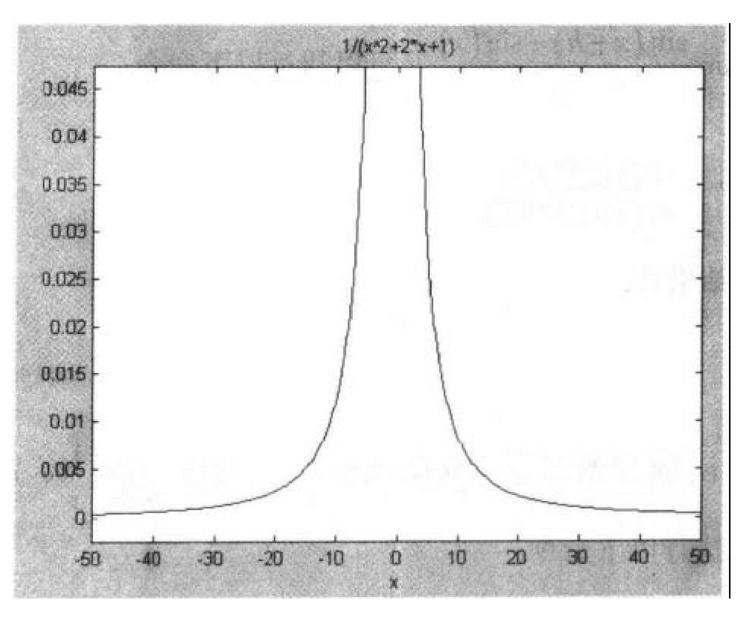



图 5-1 函数 g 的图像

### 5.1.2 符号求极限

函数的极限在高等数学中占有重要的地位。MATLAB 中提供了求极限的命令 limit，其调用格式有：

- limit(S,v)：其中 $S$ 为表达式，$v$ 为指定变量。该命令的功能：用于求当 $v\to 0$ 时表达式 $S$ 的极限值。
- limit(S)：其中 $S$ 为表达式。该命令的功能：用于求当系统的默认变量→0 时表达式 $S$ 的极限值。
- limit(F,x,a)：该命令的功能：用于求当 $x\to a$ 时表达式 $F$ 的极限值。
- limit(F,x,a,'right')：该命令的功能：用于求当 $x\to a^+$时表达式 $F$ 的极限值（即右极限值）。

● limit(F,x,a,'left')：该命令的功能：用于求当 $x \to a^-$ 时表达式 $F$ 的极限值（即左极限值）。
接下来，通过实例说明怎样求函数的极限。

【例 5-4】

（1）求：$f(x)=\lim\limits_{x \to 0} \frac{\sin x}{x}$、$g(x)=\lim\limits_{y \to 0} \sin(x+2y)$ 的极限。可以执行以下命令：

```
syms x y
f=sin(x)/x;            %表达式赋值
g=sin(x+2*y);          %表达式赋值
fx=limit(f)            %求 f(x)的极限
gx=limit(g,y,0)        %求 g(x)的极限
```

以上命令的执行结果为：

```
fx =
1
gx =
sin(x)
```

（2）求极限：$\lim\limits_{h \to 0} \frac{\sin(x+h)-\sin(x)}{h}$。可以执行以下命令：

```
syms x h
f=(sin(x+h)-sin(x))/h;   %表达式赋值
fx=limit(f,h,0)          %求 f(x)的极限
```

以上命令的执行结果为：

```
fx =
cos(x)
```

（3）求下列各函数式的极限：$f(x)=\lim\limits_{x \to 0^-} \frac{1}{x}$、$g(x)=\lim\limits_{x \to 0^+} \frac{1}{x}$、$m(x)=\lim\limits_{x \to +\infty} \left(1+\frac{a}{x}\right)^x$ 和 $n(x)=\lim\limits_{x \to +\infty} e^{-x}$。可以执行以下命令：

```
syms a x
mn=[(1+a/x)^x,exp(-x)];            %表达式赋值
fx=limit(1/x,x,0,'left')           %求 1/x 当 x--->0 时的左极限值
gx=limit(1/x,x,0,'right')          %求 1/x 当 x--->0 时的右极限值
mnx=limit(mn,x,inf,'left')         %求矩阵当 x--->inf 时的左极限值
```

以上命令的执行结果为：

```
fx =
-inf
gx =
inf
mnx =
[ exp(a),        0]
```

即 $m(x)$=exp($a$)，$n(x)$==0。

## 5.2 级数

级数也是高等数学中的一项重要内容，工程应用中更是经常遇到级数求和以及级数展开的问题，所有这些功能 MATLAB 均能实现。

### 5.2.1 级数求和

MATLAB 的级数求和功能十分强大。它主要使用 symsum 命令，其调用格式有：

- symsum(S)：以函数 findsym(S)决定的变量为自变量（比如说自变量为 $k$），求 $k$ 从 0 开始到 $k$-1 为止 $S$ 的前 $k$ 项和。
- symsum(S,v)：功能同上，只不过指定自变量为 $v$。
- symsum(S,v,a,b)：求自变量 $v$ 从 $a$ 变化到 $b$ 时 $S$ 的和。

【例 5-5】

（1）求下列级数的和：$f(k)=\sum_{k=0}^{k-1} k$、$s(n)=a\sum_{n=0}^{n-1} cc^n$、$t(n)=m\sum_{n=0}^{n-1}\cos^2(n)$。可以执行以下命令：

```
syms k a c n m
sk=simple(symsum(k))        %求 f(k)的前 k 项和并化成简单的形式
s=a*c^n;
ss=simple(symsum(s))        %求 s(n)的前 n 项和并化成简单的形式
t=m*(cos(n)^2);
st=simple(symsum(t))        %求 t(n)的前 n 项和并化成简单的形式
```

以上命令的执行结果为：

```
sk =
1/2*k*(k-1)
ss =
a*c^n/(c-1)
st =
(-2*m*n+2*m*n*cos(2)-m*cos(2*n-2)+m*cos(2*n)+m-m*cos(2))/(-4+4*cos(2))
```

（2）求 $s(q)=aq^n$ 关于 $q$ 的前 $n$ 项和。可以执行以下命令：

```
syms a q n
s=a*q^n;
ss2=simple(symsum(s,q))
```

以上命令的执行结果为：

```
ss2 =
sum(a*q^n,q)
```

该例说明：symsum 是以 $q$ 为指定的求和变量进行计算的，由于无法化简故系统只能给出 sum(a*q^n,q)这样的结果。

（3）求下列级数的和：$\sum_{k=0}^{10} k^2$、$\sum_{k=10}^{11} k^2$ 和 $\sum_{k=11}^{10} k^2$。可以执行以下命令：

```
syms k
a=symsum(k^2,0,10)
b=symsum(k^2,10,11)
c=symsum(k^2,11,10)
```

则以上命令执行结果为：

```
a =
385
b =
221
c =
0
```

由该例的计算结果可知，在级数求和中要注意变量的变化次序。$\sum_{k=11}^{10} k^2$ 本身就是错误的写法，因此 MATLAB 的计算结果为 0。

（4）试分别计算当 $m$=1、$n$=10 和 $m$=1、$n$=+∞时下列级数的和：$\sum_{x=m}^{n} \frac{a}{x}$、$\sum_{x=m}^{n} \frac{b}{x^2}$、$\sum_{x=m}^{n} \frac{c}{x^3}$ 以及 $\sum_{x=m}^{n} \frac{1}{x^2}$。可以执行以下命令：

```
syms x a b c
f=a/x;
g=b/x^2;
h=c/x^3;
p=1/x^2;
sf1=symsum(f,x,1,10)
sf2=symsum(f,x,1,inf)
sg1=symsum(g,x,1,10)
sg2=symsum(g,x,1,inf)
sh1=symsum(h,x,1,10)
sh2=symsum(h,x,1,inf)
sp1=symsum(p,x,1,10)
sp2=symsum(p,x,1,inf)
```

以上命令的执行结果为：

```
sf1 =
7381/2520*a
sf2 =
```

```
signum(a)*inf+a*eulergamma
sg1 =
1968329/1270080*b
sg2 =
1/6*b*pi^2
sh1 =
19164113947/16003008000*c
sh2 =
c*zeta(3)
sp1 =
1968329/1270080
sp2 =
1/6*pi^2
```

当然把上述程序中的语句：

```
sp1=symsum(p,x,1,10)
sp2=symsum(p,x,1,inf)
```

改为：

```
sp1=numeric(symsum(p,x,1,10))
sp2=numeric(symsum(p,x,1,inf))
```

则得到了 sp1,sp2 的浮点数表示形式：

```
sp1 =
    1.5498
sp2 =
    1.6449
```

### 5.2.2 级数展开

本节讲述泰勒级数展开、傅立叶级数展开的 MATLAB 实现。

#### 1．一元函数的泰勒级数展开

MATLAB 提供的 taylor 命令可以实现一元函数的泰勒级数展开，其调用格式为：

- taylor(f)：用于求 $f$ 关于默认变量的 5 阶近似麦克劳林多项式。
- taylor(f,n)：用于求 $f$ 关于默认变量的 $n$-1 阶近似麦克劳林多项式。
- taylor(f,v)：同上，只不过自变量为指定变量 $v$。
- taylor(f,a)：前三种调用格式求出的结果均是关于自变量等于 0 时的展式，而该命令则可以求解函数 $f$ 在自变量等于 $a$ 处的泰勒展式。

【例 5-6】

（1）试用 taylor(f)命令求 $e^{-x}$ 的泰勒展式。可以执行以下命令：

```
syms x
taylor(exp(-x))
```

以上命令的执行结果为：

```
ans =
1-x+1/2*x^2-1/6*x^3+1/24*x^4-1/120*x^5
```

（2）已知 $f=a\sin(x)y^x+u*\cos(v)$，$g=ae^v$，试用命令 taylor 求 $f$、$g$ 的泰勒展式。可以执行以下命令：

```
syms a x y u v
f=a*sin(x)*y^x+u*cos(v);
g=a*exp(v);
tf=taylor(f)
tg=taylor(g)
```

以上命令的执行结果为：

```
tf =
u*cos(v)+a*x+a*log(y)*x^2+(1/2*a*log(y)^2-1/6*a)*x^3+(1/6*a*log(y)^3-1/6*a*log(y))*x^4+(1/24*
a*log(y)^4-1/12*a*log(y)^2+1/120*a)*x^5
tg =
a+a*v+1/2*a*v^2+1/6*a*v^3+1/24*a*v^4+1/120*a*v^5
```

（3）求 $f(a)=x\cos(a)+e^a$ 的泰勒展式。可以执行以下命令：

```
syms a x
f=x*cos(a)+exp(a);
tf=taylor(f,a)
```

以上命令的执行结果为：

```
tf =
x+1+a+(1/2-1/2*x)*a^2+1/6*a^3+(1/24*x+1/24)*a^4+1/120*a^5
```

前面列举的例子所得到的结果均是按 taylor 命令默认的展开方式展开的，即关于自变量在 0 点处的 5 阶泰勒展式。接下来的实例则介绍了更一般情况下泰勒展式的求解方法。

（4）分别求函数 $f(x)=y\cos x+be^x$ 在 $x=4$ 和 $x=a$ 处的泰勒展式。可以执行以下命令：

```
syms a b x y
f=y*cos(x)+b*exp(x);
tf1=taylor(f,4)          %求 f 在 x=4 处的泰勒展式
tf2=taylor(f,a)          %求 f 在 x=a 处的泰勒展式
```

以上命令的执行结果为：

```
tf1 =
y+b+b*x+(-1/2*y+1/2*b)*x^2+1/6*b*x^3
tf2 =
y*cos(a)+b*exp(a)+(-y*sin(a)+b*exp(a))*(x-a)+(-1/2*y*cos(a)+1/2*b*exp(a))*(x-a)^2+(1/6*b*exp(a)+1/6*
y*sin(a))*(x-a)^3+(1/24*y*cos(a)+1/24*b*exp(a))*(x-a)^4+(-1/120*y*sin(a)+1/120*b*exp(a))*(x-a)^5
```

### 2．多元函数的泰勒级数展开

MATLAB 提供的 mtaylor 命令可以求多元函数的完全泰勒展式。该命令有以下几种调用格式：

- mtaylor(f,v)
- mtaylor(f,v,n)
- mtaylor(f,v,n,w)

以上式中各表示如下：

- *f*：待完全展开的代数表达式。
- *v*：变量名列表，格式为[var1=p1,var2=p2,var3=p3, …,varn=pn]。执行命令后，系统根据列表中的变量名和具体数值求解多元函数 $f$ 在点(p1,p2,p3, …,pn)处的泰勒展式。若某个变量（比如说 vari）只有变量名而没有给出具体数值，那么将视该变量的值为 0。
- *n*：泰勒展式的阶数，为一非负整数。
- *w*：与变量名列表同维的正整数列表，用于设置相应变量在展开时的权重。

注意：命令 mtaylor 并没有放在 MATLAB 符号运算工具箱的命令列表中，它是 MAPLE 符号运算函数库中的命令。mtaylor 命令的调用格式与 MATLAB 命令的调用格式也不相同。首先要把 mtaylor 命令从 MAPLE 的函数库读入工作空间，然后借助于专门用于调用 MAPLE“引擎”的函数 maple 就可以使用 mtaylor 命令了。

因此，在 MATLAB 内实现多元函数完全泰勒展式的过程为：

```
maple('readlib(mtaylor)');
maple('mtaylor(f,v,n,w)')
```

【例 5-7】

试求函数 $\sin\left(x^2+\dfrac{y^2}{z}\right)$ 在点(1,0,0)处的 3 阶泰勒展式。可以执行以下命令：

```
%从 MAPLE 库中读入 mtaylor 命令
maple('readlib(mtaylor)');

%求所给函数在点(1,0,0)处的泰勒展式
s1=maple('mtaylor(sin(x^2+y^2/z),[x=1,y=0,z=0],3)')

%功能同上，只给出变量 y,z 而不给出具体数值，则系统将视之为 0
s2=maple('mtaylor(sin(x^2+y^2/z),[x=1,y,z],3)')
```

以上命令的执行结果为：

```
s1 =
sin(1)+2*cos(1)*(x-1)+cos(1)*y^2/z+(-2*sin(1)+cos(1))*(x-1)^2-2*sin(1)*y^2/z*(x-1)-1/2*sin(1)*y^4/z^2
s2 =
```

```
sin(1)+2*cos(1)*(x-1)+cos(1)*y^2/z+(-2*sin(1)+cos(1))*(x-1)^2-2*sin(1)*y^2/z*(x-1)-1/2*sin(1)*y^4/z^2
```

可见，结果 $s1$ 与 $s2$ 完全相同。

### 3. 傅立叶级数展开

易知函数 $f(x)$ 的傅立叶级数表达式为：

$$f(x)=\frac{a_0}{2}+\sum_{n=1}^{\infty}(a_n\cos nx+b_n\sin nx)$$

其中：

$a_0=\frac{1}{\pi}\int_{-\pi}^{\pi}f(x)\mathrm{d}x$，$a_n=\frac{1}{\pi}\int_{-\pi}^{\pi}f(x)\cos nx\mathrm{d}x$，$b_n=\frac{1}{\pi}\int_{-\pi}^{\pi}f(x)\sin nx\mathrm{d}x$。式中 $n$=1,2,3, …

MATLAB 没有直接提供专用于傅立叶级数展开的命令。但是通过编程可以很方便地实现这 -功能。程序清单如下：

```
function [a0,an,bn]=mfourier(f)
% 该函数用于求取 f 的傅立叶级数展开式的系数:a0 an bn
syms n x                          %符号变量说明
a0=int(f,-pi,pi)/pi;              %计算系数 a0
an=int(f*cos(n*x),-pi,pi)/pi;     %计算系数 an
bn=int(f*sin(n*x),-pi,pi)/pi;     %计算系数 bn
```

将上述程序保存为 mfourier.m 文件，这样就创建了求函数 $f$ 的傅立叶级数展开式系数的命令 mfourier。

【例 5-8】

试求函数 $f(x)=x^2+x$ 的傅立叶级数展式。可以运行以下程序：

```
syms x
f=x^2+x;
[a0,an,bn]=mfourier(f)
```

以上程序的运行结果为：

```
a0 =
2/3*pi^2
an =
2*(n^2*pi^2*sin(pi*n)-2*sin(pi*n)+2*pi*n*cos(pi*n))/n^3/pi
bn =
-2*(-sin(pi*n)+pi*n*cos(pi*n))/n^2/pi
```

若在上述程序的后面加入以下命令：

```
pretty(a0)
pretty(an)
pretty(bn)
```

则可把 a0、an、bn 由机器格式转化为手写格式：

```
                              2
                         2/3 pi

       2    2
      n   pi   sin(pi n) - 2 sin(pi n) + 2 pi n cos(pi n)
    2 --------------------------------------------------
                             3
                            n   pi

                  -sin(pi n) + pi n cos(pi n)
              -2 ---------------------------
                             2
                            n   pi
```

事实上，手算 $f(x)$的傅立叶级数展式时就不会有这么复杂。这是因为 MATLAB 是按照符号运算的法则进行上述计算的，因此它不能把cos($n\pi$)自动化为$(-1)^n$的形式，也不能把sin($n\pi$)自动化为 0。借助于 MATLAB 的化简和替代命令把上述程序改为：

```
syms x n
f=x^2+x;
[a0,an,bn]=mfourier(f)
a0=simplify(a0)                      %用一般化简法化简 a0
an=simplify(an);                     %用一般化简法化简 an
bn=simplify(bn);                     %用一般化简法化简 bn
an=subs(an,sin(pi*n),0);             %把 an 表达式中的 sin(pi*n)用 0 替换
an=subs(an,cos(pi*n),(-1)^n)         %把 an 表达式中的 cos(pi*n)用(-1)^n 替换
bn=subs(bn,sin(pi*n),0);             %把 bn 表达式中的 sin(pi*n)用 0 替换
bn=subs(bn,cos(pi*n),(-1)^n)         %把 bn 表达式中的 cos(pi*n)用(-1)^n 替换
```

则显示结果为：

```
a0 =
2/3*pi^2
an =
2*(n^2*pi^2*sin(pi*n)-2*sin(pi*n)+2*pi*n*cos(pi*n))/n^3/pi
bn =
-2*(-sin(pi*n)+pi*n*cos(pi*n))/n^2/pi
a0 =
2/3*pi^2
an =
4/n^2*(-1)^n
bn =
-2/n*(-1)^n
```

显然后一种输出格式的输出结果要简洁得多。

## 5.3 符号矩阵的有关操作命令

在 MATLAB 中涉及符号矩阵的线性代数问题，其操作命令和 MATLAB 针对数值矩阵的有关操作命令完全相同。为了便于读者理解，本节将详细介绍符号矩阵中常用的有关操作命令。

### 5.3.1 diag 命令（求矩阵的对角线）

功能：求矩阵的对角线。

其调用格式包括：

- diag($v$,$k$)：当 $v$ 是由 $n$ 个元素组成的矢量时，该命令的返回值是阶数为 $n$+abs($k$)的方阵，其中第 $k$ 条对角线由矢量 $v$ 的元素组成，其余元素由 0 组成。当 $k$=0 时，$v$ 为主对角线；当 $k$>0 时，$v$ 位于主对角线之上；当 $k$<0 时，$v$ 位于主对角线之下。
- diag($v$)：与 diag($v$,0)完全相同，把矢量 $v$ 置于主对角线上。
- diag($A$,$k$)：其中 $A$ 为矩阵。该命令返回值是由矩阵 $A$ 的第 $k$ 条对角线的元素所组成的列矢量。
- diag($A$)：相当于 diag($A$,0)，得到由矩阵 $A$ 的主对角线元素所组成的列矢量。

【例 5-9】

（1）已知矢量 $v$=[$a$ $b$ $c$]，矩阵 $x=\begin{bmatrix}1 & a & 2\\ b & 3 & 4\\ 5 & c & 6\end{bmatrix}$，执行以下命令：

```
syms a b c            %变量说明
v=[a b c];            %v 是由 a b c 组成的行矢量（长度为 3）
a1=diag(v,1)          %得到一个 3+1=4 阶的矩阵 a1，其中主对角线之上的第 1 对角线
                      %由 v 的元素组成
a2=diag(v,0)          %得到一个 3+0=3 阶的矩阵 a2，其主对角线由矢量 v 组成
a3=diag(v)            %与 diag(v,0) 的功能完全相同
x=[1 a 2
   b 3 4
   5 c 6];            %x 为一符号矩阵
b1=diag(x,1)          %b1 是一个列矢量，其元素由矩阵 x 的第一对角线组成
b2=diag(x,0)          %b2 是一个列矢量，其元素由矩阵 x 的主对角线组成
b3=diag(x,-1)         %b3 是一个列矢量，其元素由位于矩阵 x 主对角线下的第一对角
                      %线组成
```

以上命令的执行结果为：

```
a1 =
[ 0, a, 0, 0]
[ 0, 0, b, 0]
[ 0, 0, 0, c]
[ 0, 0, 0, 0]
a2 =
```

```
[ a, 0, 0]
[ 0, b, 0]
[ 0, 0, c]
a3 =
[ a, 0, 0]
[ 0, b, 0]
[ 0, 0, c]
b1 =
[ a]
[ 4]
b2 =
[ 1]
[ 3]
[ 6]
b3 =
[ b]
[ c]
```

由显示的结果可知，*a*1~*a*3 均是符号阵、*b*1~*b*3 均是符号矢量。

（2）利用 diag 命令可以构造大矩阵，例如执行命令：

```
m=5;
diag(-m:m)+diag(ones(2*m,1),1)+diag(ones(2*m,1),-1)
```

显示结果为：

```
ans =
    -5    1    0    0    0    0    0    0    0    0    0
     1   -4    1    0    0    0    0    0    0    0    0
     0    1   -3    1    0    0    0    0    0    0    0
     0    0    1   -2    1    0    0    0    0    0    0
     0    0    0    1   -1    1    0    0    0    0    0
     0    0    0    0    1    0    1    0    0    0    0
     0    0    0    0    0    1    1    1    0    0    0
     0    0    0    0    0    0    1    2    1    0    0
     0    0    0    0    0    0    0    1    3    1    0
     0    0    0    0    0    0    0    0    1    4    1
     0    0    0    0    0    0    0    0    0    1    5
```

### 5.3.2 triu 命令（抽取矩阵的上三角部分）

功能：抽取矩阵的上三角部分组成一个新的矩阵，其余元素用 0 填充。

其调用格式有：

- triu(A)：抽取矩阵的上三角部分组成一个新的矩阵，其余元素用 0 填充。
- triu(A,k)：抽取矩阵的第 *k* 条对角线以上的三角部分组成一个新的矩阵，其余元素用 0 填充。当 *k*=0 时，triu(A,0)与 triu(A)功能完全相同，抽取矩阵 *A* 主对角线以上的三角部分；*K*>0 抽取的元素对应矩阵 *A* 主对角线以上、第 *k* 条对角线以上的部分；*K*<0

抽取的元素对应矩阵 $A$ 主对角线以下、第 $k$ 条对角线以上的部分。

【例 5-10】

已知矩阵 $A=\begin{bmatrix} a & 1 & 2 & 3 \\ b & a+b & 3 & 4 \\ c & d & b+c & d \\ a & b & d & 5 \end{bmatrix}$，执行以下命令：

```
syms a b c d
A=[a 1    2    3
   b a+b 3    4
   c d    b+c d
   a b    d    5];
a1=triu(A)                  %抽取矩阵 A 主对角线以上的部分组成矩阵 a1 的上三角部
                            %分，其余元素补 0
a2=triu(A,0)                %功能完全和上条命令一样
a3=triu(A,1)                %从 A 的第一对角线开始抽取 A 的上三角部分组成矩阵 a3，
                            %其余元素补 0
a4=triu(A,-1)               %从 A 主对角线下的第一对角线开始抽取 A 的上三角部分
                            %组成矩阵 a4，其余元素补 0
```

则得到的显示结果为：

```
a1 =
[   a,   1,   2,   3]
[   0, a+b,   3,   4]
[   0,   0, b+c,   d]
[   0,   0,   0,   5]
a2 =
[   a,   1,   2,   3]
[   0, a+b,   3,   4]
[   0,   0, b+c,   d]
[   0,   0,   0,   5]
a3 =
[ 0, 1, 2, 3]
[ 0, 0, 3, 4]
[ 0, 0, 0, d]
[ 0, 0, 0, 0]
a4 =
[   a,   1,   2,   3]
[   b, a+b,   3,   4]
[   0,   d, b+c,   d]
[   0,   0,   d,   5]
```

### 5.3.3 tril 命令（抽取矩阵的下三角部分）

功能：抽取矩阵的下三角部分组成一个新的矩阵，其余元素用 0 填充。

其调用格式有：

- tril(A)：抽取矩阵的下三角部分组成一个新的矩阵，其余元素用 0 填充。
- tril(A,k)：抽取矩阵的第 $k$ 条对角线以下的三角部分组成一个新的矩阵，其余元素用 0 填充。当 $k$=0 时，triu(A,0)与 triu(A)功能完全相同：抽取矩阵 $A$ 主对角线以下的三角部分；$k$>0 抽取的元素对应矩阵 $A$ 主对角线之上、第 $k$ 条对角线以下的部分；$k$<0 抽取的元素对应矩阵 $A$ 主对角线以下、第 $k$ 条对角线以下的部分。

【例 5-11】

已知矩阵 $A=\begin{bmatrix} a & 1 & 2 & 3 \\ b & a+b & 3 & 4 \\ c & d & b+c & d \\ a & b & d & 5 \end{bmatrix}$，执行以下程序：

```
syms a b c d
A=[a 1    2    3
   b a+b 3    4
   c d    b+c d
   a b    d    5];
a1=tril(A)              %抽取矩阵 A 主对角线以下的部分组成矩阵 a1 的下三角部分，
                        %其余元素补 0
a2=tril(A,0)            %功能完全和上条命令一样
a3=tril(A,1)            %从 A 的第一对角线开始抽取 A 的下三角部分组成矩阵 a3，其
                        %余元素补 0
a4=tril(A,-1)           %从 A 主对角线下的第一对角线开始抽取 A 的下三角部分组成
                        %矩阵 a4，其余元素补 0
```

显示结果为：

```
a1 =
[   a,   0,   0,   0]
[   b, a+b,   0,   0]
[   c,   d, b+c,   0]
[   a,   b,   d,   5]
a2 =
[   a,   0,   0,   0]
[   b, a+b,   0,   0]
[   c,   d, b+c,   0]
[   a,   b,   d,   5]
a3 =
[   a,   1,   0,   0]
[   b, a+b,   3,   0]
[   c,   d, b+c,   d]
[   a,   b,   d,   5]
a4 =
[ 0, 0, 0, 0]
[ b, 0, 0, 0]
[ c, d, 0, 0]
[ a, b, d, 0]
```

### 5.3.4 inv 命令（矩阵求逆）

功能：矩阵求逆。

其调用格式为：

● inv(A)：返回矩阵 $A$ 的逆。

【例 5-12】

已知矩阵 $A=\begin{bmatrix} a & 1 & 2 \\ b & a & 3 \\ 4 & d & c \end{bmatrix}$，执行以下程序：

```
syms a b c d
A=[a 1 2
   b a 3
   4 d c];
a1=inv(A)        %求矩阵 A 的逆
pretty(a1)       %将矩阵 a1 由机器格式转化为手写格式
```

则显示结果为：

```
a1 = .
[    (a*c-3*d)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a),     (-c+2*d)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a),    -(-3+2*a)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a)]
[    -(b*c-12)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a),     (a*c-8)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a),   -(3*a-2*b)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a)]
[ -(-b*d+4*a)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a),     -(a*d-4)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a),     (a^2-b)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a)]
```

```
[ a c - 3 d      -c + 2 d       -3 + 2 a ]
[ ---------      --------      - -------- ]
[    %1             %1              %1    ]
[                                         ]
[   b c - 12      a c - 8       3 a - 2 b ]
[ - --------      -------      - ---------]
[     %1            %1              %1    ]
[                                         ]
[                                2        ]
[  -b d + 4 a     a d - 4       a  - b    ]
[- ----------    - ------      - ------   ]
[     %1            %1            %1      ]

                2
        %1 := a  c - 3 a d - b c + 2 b d + 12 - 8 a
```

显然手写格式看起来要舒服些。

### 5.3.5 det 命令（求矩阵的行列式）

功能：求矩阵的行列式。

其调用格式为：

● det(A)：返回矩阵 $A$ 的行列式。

【例 5-13】

已知矩阵 $A=\begin{bmatrix} a & 1 & 2 \\ b & a & 3 \\ 4 & d & c \end{bmatrix}$，执行以下程序：

```
syms a b c d
A=[a 1 2
   b a 3
   4 d c];
a1=det(A)          %求矩阵 A 的行列式
pretty(a1)      %将表达式 a1 由机器格式转化为手写格式
```

则显示结果为：

```
a1 =
a^2*c-3*a*d-b*c+2*b*d+12-8*a
                          2
                         a  c - 3 a d - b c + 2 b d + 12 - 8 a
```

可见手写式较为直观。

### 5.3.6 rank 命令（求矩阵的秩）

功能：求矩阵的秩。

其调用格式有：

● rank(A,tol)：返回矩阵 $A$ 的奇异值中大于误差 tol 的奇异值个数。

● rank(A)：同上，默认精度 tol=max(size(A)*norm(A))*eps。

【例 5-14】

已知矩阵 $A=\begin{bmatrix} a & 1 & 2 & b \\ b & a & 3 & c \\ 4 & d & c & d \end{bmatrix}$，执行以下程序：

```
syms a b c d
A=[a 1 2 b
   b a 3 c
   4 d c d];
a=rank(A)          %求矩阵 A 的秩
```

则显示结果为：

```
a =
3
```

### 5.3.7 rref 命令（求矩阵的缩减行阶梯矩阵）

功能：求矩阵的缩减行阶梯矩阵。

其调用格式有以下几种：

- *R*=rref(*A*)：返回 *A* 矩阵的缩减行阶梯矩阵 *R*，计算过程使用的是高斯-约当消元法和行主元法。
- [*R*,*jb*]=rref(*A*)：返回 *R* 同上，返回值 *jb* 为一矢量。这种算法的思想是：矩阵 *A* 的秩 *r*=length(*jb*)，而 *x*(*jb*)为线性系统 *Ax*=*b* 的边界变量；*A*(:,*jb*)为 *A* 范围内的基；*R*(1:*r*,*jb*)为 *r*×*r* 阶的不确定性矩阵。
- [*R*,*jb*]=rref(*A*,tol)：同上，tol 为指定误差，即参数 tol 用来确定什么时候元素可以忽略不计。

【例 5-15】

已知矩阵 $A=\begin{bmatrix} a & 1 & 2 & 4 \\ b & a & 3 & d \\ 4 & d & c & 5 \end{bmatrix}$，执行以下程序：

```
syms a b c d
A=[a 1 2 4
   b a 3 d
   4 d c 5];
a1=rref(A)       %求矩阵 A 的缩减行阶梯矩阵
pretty(a1)       %将表达式 a1 由机器格式转化为手写格式
```

则显示结果为：

```
a1=
[1,0,0,     (4*a*c-10*a-c*d-12*d+15+2*d^2)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a)]
[0,1,0,     (a*c*d-15*a+48-4*b*c+10*b-8*d)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a)]
[0,0,1,  (5*a^2-5*b-16*a-a*d^2+4*b*d+4*d)/(a^2*c-3*a*d-b*c+2*b*d+12-8*a)]

          [                                        2]
          [            4 a c - 10 a - c d - 12 d + 15 + 2 d ]
          [1    0    0   ------------------------------------    ]
          [                         %1                         ]
          [                                                    ]
          [            a c d - 15 a + 48 - 4 b c + 10 b - 8 d]
          [0    1    0   ------------------------------------    ]
          [                         %1                         ]
          [                                                    ]
          [              2              2                      ]
          [            5 a  - 5 b - 16 a - a d  + 4 b d + 4 d]
          [0    0    1   ------------------------------------    ]
          [                          %1                        ]
```

```
                 2
%1 := a   c - 3 a d - b c + 2 b d + 12 - 8 a
```

对该例的显示结果来说，机器格式和手写格式相差不多，均比较简洁。

### 5.3.8 null 命令（求矩阵的零空间的正交基）

功能：求矩阵的零空间的正交基。

其调用格式有以下两种：

- Z=null(*A*)：求矩阵 *A* 的零空间的正交基，它是由矩阵 *A* 的奇异值分解得到的。
- Z=null(*A*,'*r*')：求矩阵 *A* 的零空间的正交基，它是由缩减行阶梯矩阵得到的并且 *A**Z=0。

【例 5-16】

（1）已知矩阵 $A=\begin{bmatrix}1 & 2 & 3\\1 & 2 & 3\\1 & 2 & 3\end{bmatrix}$，执行以下程序：

```
A=[1 2 3;1 2 3;1 2 3];
Z1=null(A)
Z2=null(A,'r')
test=A*Z2
```

则显示结果为：

```
Z1 =
    0.9636         0
   -0.1482   -0.8321
   -0.2224    0.5547
Z2 =
    -2    -3
     1     0
     0     1
test =
     0     0
     0     0
     0     0
```

（2）已知矩阵 $A=\begin{bmatrix}a & 1 & 2 & 4\\b & a & 3 & d\\4 & d & c & 5\end{bmatrix}$，执行以下程序：

```
syms a b c d
A=[a 1 2 4
   b a 3 d
   4 d c 5];
Z1=null(A)
```

则显示结果为：

```
Z1 =
[                                                                    1]
[    (a*c*d-15*a+48-4*b*c+10*b-8*d)/(4*a*c-10*a-c*d-12*d+15+2*d^2)]
[ (5*a^2-5*b-16*a-a*d^2+4*b*d+4*d)/(4*a*c-10*a-c*d-12*d+15+2*d^2)]
[   -(a^2*c-3*a*d-b*c+2*b*d+12-8*a)/(4*a*c-10*a-c*d-12*d+15+2*d^2)]
```

### 5.3.9 colspace 命令（求矩阵的列空间的基）

功能：求矩阵的列空间的基。

其调用格式为：

- Z=colspace(A)：返回矩阵 *A* 的列空间的基，并且有 size(Z,2)=rank(A)。

【例 5-17】

执行以下程序：

```
A=colspace(sym(magic(4)))  %先将 4 阶魔方阵转化为符号矩阵，再求其列空间的基
test1=size(A,2);           %求矩阵 A 的列数
test2=rank(sym(magic(4))); %求 4 阶魔方符号矩阵的秩
result=test1==test2        %判断 test1 和 test2 是否相等，并把逻辑结果赋予 result
```

则显示结果为：

```
A =
[  0,  1,  0]
[  0,  0,  1]
[  1,  0,  0]
[ -3,  1,  3]
result =
     1
```

### 5.3.10 eig 命令（求矩阵的特征值和特征矢量）

功能：求矩阵的特征值和特征矢量。

其调用格式有：

- *E*=eig(*X*)：返回由方阵 *X* 的特征值组成的矢量。
- [*V*,*D*]=eig(*X*)：返回方阵 *X* 的特征值矩阵 *D* 和特征矢量矩阵 *V*，其中 *X*、*V*、*D* 之间满足 *XV*=*VD*；特征值矩阵 *D* 是以 *X* 的特征值为对角线的元素生成的对角阵；矩阵 *X* 的第 *k* 个特征值对应的特征矢量是矩阵 *D* 的第 *k* 列列矢量，只有这样才有 *XV*=*VD*。

【例 5-18】

已知矩阵 $X=\begin{bmatrix} a & b \\ c & d \end{bmatrix}$，试求 *X* 的特征矢量、特征值矩阵以及特征矢量阵。可以执行以下命令：

```
syms a b c d
X=[a b;c d];
E1=eig(X)              %E1 是由矩阵 X 的特征值所组成的矢量
```

```
[V1,D1]=eig(X)        %V1 是矩阵 X 的特征值矩阵，D1 是矩阵 X 的特征矢量矩阵
```

则得到的执行结果为：

```
E1 =
[ 1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2)]
[ 1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2)]
V1 =
[-(-1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2))/c, -(-1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*b*c)^ (1/ 2))
/c]
[                          1,                                              1]
D1 =
[1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2),                              0]
[                          0,         1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2)]
```

### 5.3.11 svd 命令（矩阵的奇异值分解）

功能：矩阵的奇异值分解。

其调用格式为：

- $S$=svd(X)：返回由矩阵 $X$ 的奇异值组成的矢量。

【例 5-19】

已知矩阵 $X=\begin{bmatrix} a & b \\ c & d \end{bmatrix}$，试对 $X$ 进行奇异值分解。可以执行以下命令：

```
syms a b c d
X=[a b;c d];
s1=svd(X)        %s1 是由矩阵 X 的奇异值所组成的矢量
```

以上命令的执行结果为：

```
s1 =
[ 1/2*(2*a*conj(a)+2*d*conj(d)+2*b*conj(b)+2*c*conj(c)+2*(a^2*conj(a)^2-2*a*conj(a)*d*conj(d)+2 *a*
conj(a)*b*conj(b)+2*a*conj(a)*c*conj(c)+d^2*conj(d)^2+2*d*conj(d)*b*conj(b)+2*d*conj(d)*c*conj(c)+b
^2*conj(b)^2-2*b*conj(b)*c*conj(c)+c^2*conj(c)^2+4*a*conj(c)*d*conj(b)+4*b* conj(d)*c*conj(a))^
(1/2))^(1/2)]
[ 1/2*(2*a*conj(a)+2*d*conj(d)+2*b*conj(b)+2*c*conj(c)-2*(a^2*conj(a)^2-2*a*conj(a)*d*conj(d)+   2*a
*conj(a)*b*conj(b)+2*a*conj(a)*c*conj(c)+d^2*conj(d)^2+2*d*conj(d)*b*conj(b)+2*d*conj(d)*c*conj(c)+
b^2*conj(b)^2-2*b*conj(b)*c*conj(c)+c^2*conj(c)^2+4*a*conj(c)*d*conj(b)+4*b*conj(d)*c*conj(a))
^(1/2))^(1/2)]
```

### 5.3.12 jordan 命令（求矩阵的约当标准形）

功能：返回矩阵的约当标准形。

其调用格式为：

- jordan(X)：返回矩阵 $X$ 的约当标准形。
- $[V,J]$= jordan(X)：除了返回矩阵 $X$ 的约当标准形 $J$ 外，还给出了相似变换阵 $V$，并有 $V\backslash A*V=J$。

【例 5-20】

已知矩阵 $X=\begin{bmatrix} a & b \\ c & d \end{bmatrix}$，试求 X 的约当标准形。可以执行以下命令：

```
syms a b c d
X=[a b;c d];
s1=jordan(X)     %返回矩阵 X 的约当标准形 s1
[V,J]=jordan(X)  %返回矩阵 X 的约当标准形 J，返回相似变换阵 V
```

则得到的执行结果为：

```
s1 =
[1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2),                                    0]
[                              0,              1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2)]
J =
[1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2),                              0]
[                          0,            1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2)]
V =
[ -1/2*(2*a-2*d+abs(a^2-2*a*d+d^2+4*c*b)^(1/2)+abs(a^2-2*a*d+d^2+4*c*b)^(1/2)*signum(a^2-2*a*d+d^
2+4*c*b)+i*abs(a^2-2*a*d+d^2+4*c*b)^(1/2)-i*abs(a^2-2*a*d+d^2+4*c*b)^(1/2)*signum(a^2-2*a*d+d^2+
4*c*b))/abs(a^2-2*a*d+d^2+4*c*b)^(1/2)/(-1-signum(a^2-2*a*d+d^2+4*c*b)-i+i*signum(a^2-2*a*d+d^2+
4*c*b)),   1/2*(2*a-2*d-abs(a^2-2*a*d+d^2+4*c*b)^(1/2)-abs(a^2-2*a*d+d^2+4*c*b)^(1/2)*signum(a^2-
2*a*d+d^2+4*c*b)-i*abs(a^2-2*a*d+d^2+4*c*b)^(1/2)+i*abs(a^2-2*a*d+d^2+4*c*b)^(1/2)*signum(a^2-
2*a*d+d^2+4*c*b))/abs(a^2-2*a*d+d^2+4*c*b)^(1/2)/(-1-signum(a^2-2*a*d+d^2+4*c*b)-i+i*signum(a^2-
2*a*d+d^2+4*c*b))]
[-2*c/abs(a^2-2*a*d+d^2+4*c*b)^(1/2)/(-1-signum(a^2-2*a*d+d^2+4*c*b)-i+i*signum(a^2-2*a*d+d^2+
4*c*b)),2*c/abs(a^2-2*a*d+d^2+4*c*b)^(1/2)/(-1-signum(a^2-2*a*d+d^2+4*c*b)-i+i*signum(a^2-2*a*d
+d^2+4*c*b))]
```

### 5.3.13 poly 命令（求矩阵的特征多项式）

功能：返回矩阵的特征多项式。

其调用格式为：

- P=poly(X)：若 X 为 $n \times n$ 的矩阵，则该命令返回 X 的特征多项式 P。P 为包含 $n+1$ 个元素的矢量，是特征多项式的系数。

【例 5-21】

已知矩阵 $X=\begin{bmatrix} a & b \\ c & d \end{bmatrix}$，试求 X 的特征多项式。可以执行以下命令：

```
syms a b c d
X=[a b;c d];
P=poly(X)     %返回矩阵 X 的特征多项式
```

以上命令的执行结果为：

```
P =
x^2-x*d-a*x+a*d-b*c
```

### 5.3.14　expm 命令（求矩阵的指数形式）

功能：求矩阵的指数形式。

其调用格式为：

● expm(X)：用 pade 法计算 $e^X$。

【例 5-22】

已知矩阵 $X=\begin{bmatrix} a & b \\ c & d \end{bmatrix}$，试求 $e^X$。可以执行以下命令：

```
syms a b c d
X=[a b;c d];
P=expm(X)   %计算矩阵 X 的指数
```

以上命令的执行结果为：

```
P =
[ -1/2*(-(a^2-2*a*d+d^2+4*c*b)^(1/2)*exp(1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2))+a*exp
(1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2))-a*exp(1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2))-
d*exp(1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2))+d*exp(1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*c*b)
^(1/2))-(a^2-2*a*d+d^2+4*c*b)^(1/2)*exp(1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2)))/(a^2-2*a*d+
d^2+4*c*b)^(1/2),-b*(exp(1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2))-exp(1/2*a+1/2*d+1/2*(a^2-
2*a*d+d^2+4*c*b)^(1/2)))/(a^2-2*a*d+d^2+4*c*b)^(1/2)]
[-c*(exp(1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2))-exp(1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*c*b)
^(1/2)))/(a^2-2*a*d+d^2+4*c*b)^(1/2),    1/2*((a^2-2*a*d+d^2+4*c*b)^(1/2)*exp(1/2*a+1/2*d+1/2*(a^2-
2*a*d+d^2+4*c*b)^(1/2))-d*exp(1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2))+d*exp
(1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2))+a*exp(1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2))-
a*exp(1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2))+(a^2-2*a*d+d^2+4*c*b)^(1/2)*exp(1/2*a+1/2*d-
1/2*(a^2-2*a*d+d^2+4*c*b)^(1/2)))/(a^2-2*a*d+d^2+4*c*b)^(1/2)]
```

注意：本节介绍的 14 个命令主要是针对符号矩阵的运算。事实上，这些命令在数值型矩阵的运算中其调用形式要丰富得多，功能也要强大得多。之所以要在符号运算这一章讲述这些内容，是为了让读者知道 MATLAB 的符号处理功能是十分强大的。MATLAB 不但具有十分强大的数值运算能力，而且它还有丰富的符号运算功能。

## 5.4　符号方程（组）的求解

利用 MATLAB 提供的有关命令不但可以方便地求解涉及符号的代数方程（组），而且还可以求解形形色色的常微分方程（组）。本节将详细介绍 MATLAB 中涉及符号的（常微分）方程（组）的求解问题。

### 5.4.1　一般代数方程（组）的求解

利用 MATLAB 提供的 solve 命令可以方便地求解涉及符号的一般代数方程（组）。其调用格式有：

- solve(Equ)：*Equ* 为符号方程，该命令可以求 *Equ* 关于系统默认变量为自变量的符号方程的解。
- solve(Equ,var)：同上，但 *var* 为指定的自变量，求出的解是关于指定变量的解。
- solve('equ1','equ2',···, 'equn')
- [a1,a2,···,an]= solve('equ1','equ2',···, 'equn','var1,var2,···,varn')

最后两种调用格式相差无几，都是求代数方程组的解，只不过最后一种调用格式指定了自变量 *var*1,*var*2,···,*varn*。事实上，*var*1,*var*2,···,*varn* 可有可无，这是因为 solve 命令只有当方程的数目和自变量的数目相同时才能进行求解，并且解得的结果并不是按照 solve 命令括号中 *var*1,*var*2,···,*varn* 的顺序分别赋给 *a*1,*a*2,···,*an* 的，而是按照英文字母表的顺序依次赋给 *a*1,*a*2,···,*an*。即当 *var*1 在所有变量中按字母表顺序排序时排在最后一个，那么在结果中 *an* 才是对应变量 *var*1 的解。

【例 5-23】

（1）求方程 $ax^2+bx+c=0$ 的解。可以执行以下命令：

```
f=sym('a*x^2+b*x+c=0');
xf=solve(f)                %求方程 f(x)=0 的根
pretty(xf)                 %将所求得的解化为手写式
```

以上命令的执行结果为：

```
xf =
[ 1/2/a*(-b+(b^2-4*a*c)^(1/2))]
[ 1/2/a*(-b-(b^2-4*a*c)^(1/2))]
                    [            2         1/2]
                    [      -b + (b   - 4 a c)   ]
                    [1/2   --------------------  ]
                    [              a              ]
                    [                             ]
                    [            2         1/2]
                    [      -b - (b   - 4 a c)   ]
                    [1/2   --------------------  ]
                    [              a              ]
```

显然手写格式看起来要简洁些。

（2）分别求方程 sin*x*+*b*tan*a*=0 及 *a*sin4+5tan(*a*+3)=0 的解，它们的自变量均为 *a*。可以执行以下命令：

```
f=sym('sin(x)+b*tan(a)=0');
g=sym('a*sin(4)+5*tan(a+3)=0');
xf=solve(f,'a')                %求方程 f(x)=0 的根
xg=solve(g,'a')                %求方程 g(x)=0 的根
```

以上命令的执行结果为：

```
xf =
-atan(sin(x)/b)
```

```
xg =
-3.4854514938227066852523169031753
```

（3）求非线性方程组$\begin{cases} a+b+x=y \\ 2ax-by=-1 \\ (a+b)^2=x+y \\ ay+bx=4 \end{cases}$的解。可以执行以下命令：

```
e1=sym('a+b+x=y');
e2=sym('2*a*x-b*y=-1');
e3=sym('(a+b)^2=x+y');
e4=sym('a*y+b*x=4');
[a,b,x,y]=solve(e1,e2,e3,e4);        %得到线性方程组的解
a=numeric(a)                         %将 a 化为数值型
b=numeric(b)                         %将 b 化为数值型
x=numeric(x)                         %将 x 化为数值型
y=numeric(y)                         %将 y 化为数值型
```

以上命令的执行结果为：

```
a =
    1.0000
   23.6037
    0.2537 - 0.4247i
    0.2537 + 0.4247i
b =
    1.0000
  -23.4337
   -1.0054 - 1.4075i
   -1.0054 + 1.4075i
x =
    1.0000
   -0.0705
   -1.0203 + 2.2934i
   -1.0203 - 2.2934i
y =
    3.0000
    0.0994
   -1.7719 + 0.4611i
   -1.7719 - 0.4611i
```

☞ 提示：可见所给的方程组共有四组解，其中两组为实数解，两组为虚数解。一般来说，用 solve 命令得到的解是精确的符号表达式，显得很不直观，通常要把所得的解化为数值型以使结果显得直观、简洁。

（4）求非线性方程组 $\begin{cases} \dfrac{u^2}{a}+v^2=0 \\ u+v=1 \\ a^2+5a=9 \end{cases}$ 的解。可以执行以下命令：

```
[a,u,v]=solve('u^2/a+v^2=0','u+v=1','a^2+5*a=9');
a=numeric(a)                    %将 a 化为数值型
u=numeric(u)                    %将 u 化为数值型
v=numeric(v)                    %将 v 化为数值型
```

以上命令的执行结果为：

```
a =
    1.4051
    1.4051
   -6.4051
   -6.4051
u =
   0.5842 - 0.4929i
   0.5842 + 0.4929i
   0.7168
   1.6532
v =
   0.4158 + 0.4929i
   0.4158 - 0.4929i
   0.2832
  -0.6532
```

共有四组解满足所给的非线性方程组。

（5）解超越方程组：$\begin{cases} x^x=4 \\ xy+y=1 \end{cases}$，可以执行以下命令：

```
[x,y]=solve('x^x=2','x*y+y=1')
```

则以上命令的执行结果为：

```
x =
log(2)/lambertw(log(2))
y =
lambertw(log(2))/(lambertw(log(2))+log(2))
```

提示：所给出的解中 lambertw 是个函数（称为 lambert $W$ 函数），lambertw(A)是指满足 $we^w=A$ 这样的表达式所对应的值。

注意：当用符号表达式 $s1,s2,\cdots,sn$ 代替 solve 命令中的符号方程组 $equ1,equ2,\cdots,equn$

时，就意味着所求的是以 $s1=0,s2=0,\cdots,sn=0$ 所构成的方程组的解。

【例 5-24】

试求当$\begin{cases} s1 = x^2 + 4xy + z \\ s2 = x + 3yz - 3 \\ s3 = y + \sin z \end{cases}$ 均为 0 时 $x$、$y$ 和 $z$ 的值。可以执行以下命令：

```
syms x y z
s1=x^2+4*x*y+z;
s2=x+3*y*z-3;
s3=y+sin(z);
[x,y,z]=solve(s1,s2,s3)
```

以上命令的执行结果：

```
x =
-1.2488907023777540227165494966924
y =
-.39875957366860047824596051700207
z =
-3.5517564826409285761070228228627
```

利用 numreic 命令可以得到 $x,y,z$ 只有四位小数的表示形式。
在上面的源代码末尾再加入以下执行命令：

```
x=numeric(x)
y=numeric(y)
z=numeric(z)
```

则所得到的执行结果为：

```
x =
    -1.2489
y =
    -0.3988
z =
    -3.5518
```

### 5.4.2 线性代数方程（组）的求解

上节介绍的 solve 命令其使用范围为所有的代数方程组。作为特例，当所给的方程组为线性方程组时，利用 MATLAB 提供的 linsolve 命令求解要方便得多。也就是说，linsolve 命令可以求解形如 $AX=B$ 的线性方程组。但该命令对矩阵 $A$ 有严格的限制：$A$ 必须是满秩的。linsolve 命令的调用格式为：

- X=linsolve(A,B)：求 $AX=B$ 的解，返回 $X$。

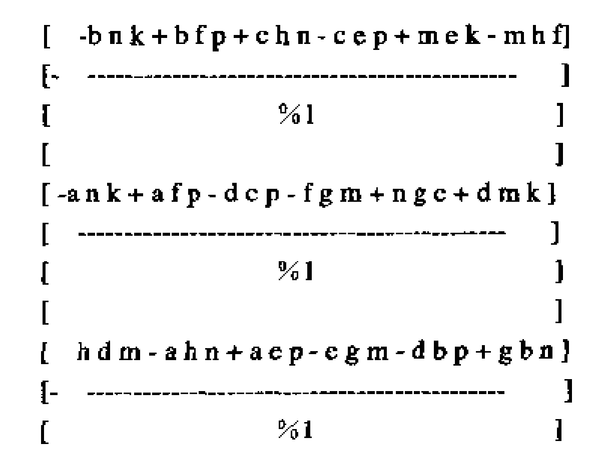
【例 5-25】

求解线性系统$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & k \end{bmatrix} X = \begin{bmatrix} m \\ n \\ p \end{bmatrix}$。可以执行以下命令：

```
syms a b c d e f g h k m n p
A=[a b c;d e f;g h k];
B=[m;n;p];
X=linsolve(A,B)              %得到 AX=B 的解 X
pretty(X)                    %将 X 化为手写格式
```

则得到的执行结果为：

```
X =
[ -(c*h*n-m*h*f+k*e*m-k*b*n+p*b*f-p*e*c)/(-g*b*f+g*e*c+a*h*f-e*a*k+b*d*k-d*h*c)]
[ (-a*k*n+a*p*f+g*c*n+k*d*m-g*m*f-p*d*c)/(-g*b*f+g*e*c+a*h*f-e*a*k+b*d*k-d*h*c)]
[ -(e*a*p-g*e*m+g*b*n-b*d*p-a*h*n+d*h*m)/(-g*b*f+g*e*c+a*h*f-e*a*k+b*d*k-d*h*c)]
```



```
[   -b n k + b f p + c h n - c e p + m e k - m h f]
[-  --------------------------------------------- ]
[                     %1                          ]
[                                                 ]
[ -a n k + a f p - d c p - f g m + n g c + d m k  ]
[  ---------------------------------------------  ]
[                     %1                          ]
[                                                 ]
[  h d m - a h n + a e p - c g m - d b p + g b n  ]
[-  --------------------------------------------- ]
[                     %1                          ]

%1 := -a e k + a h f + d b k - h d c + e g c - g b f
```

显然手写格式的解要直观一些。

### 5.4.3 常微分方程（组）的求解

MATLAB 的符号工具箱中提供了求解常微分方程（组）的命令：dsolve，其功能十分强大：不但可以求解无附加条件的常微分方程，而且可以求解有附加条件的常微分方程，并且可以求解齐次常微分方程（组）以及非齐次常微分方程（组）。接下来，将详细介绍 dsolve 命令的使用方法。

在介绍 dsolve 命令的使用方法之前，要注意的是：平时以 $y''+2y'=x$ 形式出现的常微分方程，在 MATLAB 中需要重新改写。MATLAB 中用 $D$ 表示对变量求导数，$Dy$ 表示对 $y$ 求一阶导数，$Dny$ 表示对 y 求 $n$ 阶导数。因此，$y''+2y'=x$ 这一微分方程在 MATLAB 中需描述为：$D2y+2Dy=x$。

#### 1．求解无附加条件的常微分方程

用 dsolve 命令求出的常微分方程的解就是该常微分方程的通解。dsolve 命令的调用格式

有：

- dsolve('equ')
- dsolve('equ','var')

上述命令调用格式中，*equ* 为待求解的常微分方程；第 1 种调用格式视变量 *t* 为自变量进行求解；第 2 种调用格式中 *var* 为指定变量，dsolve 将以 *var* 为自变量进行常微分方程的求解。

【例 5-26】

（1）求解常微分方程：$\frac{dy}{dx}=-ax$。可以执行以下命令：

```
y=dsolve('Dy+a*x=0','x')
```

则以上命令的执行结果为：

```
y =
-1/2*a*x^2+C1
```

其中 *C*1 表示所求出的解为通解。

若一时粗心，把执行命令写为：

```
y=dsolve('Dy+a*x=0')
```

则所得到的执行结果为：

```
y =
-a*x*t+C1
```

显然结果相差甚远。

（2）求解常微分方程：$\frac{d^2y}{dx^2}+2x=2y$ 。可以执行以下命令：

```
y=dsolve('D2y+2*x=2*y','x')
```

以上命令的执行结果为：

```
y =
x+C1*sinh(2^(1/2)*x)+C2*cosh(2^(1/2)*x)
```

**2．求解有初始条件的常微分方程**

求解有初始条件的常微分方程也比较简单：在上面的 dsolve 命令中加入初始条件就行了。此时 dsolve 命令的调用格式有：

- dsolve('equ','condition1,condition2,···,conditionn','var')
- dsolve('equ','condition1','condition2',···,'conditionn','var')

以上两种调用格式所得结果完全相同。其中：*equ* 为常微分方程；condition1，condition2，···，conditionn 为初始条件；*var* 为指定变量。

【例 5-27】

（1）求解常微分方程：$\frac{d^2y}{dx^2}+2x=2y$，且满足：$y(2)=5, y'(1)=2$。可以执行以下命令：

```
y=dsolve('D2y+2*x=2*y','y(2)=5','Dy(1)=2','x')
```

以上命令的执行结果为：

```
y =
x+1/2*(exp(2^(1/2))^4*2^(1/2)-6*exp(2^(1/2))^3+6*exp(2^(1/2))+2^(1/2))/(exp(2^(1/2))^2+1)
/exp(2^(1/2))*sinh(2^(1/2)*x)-1/2*(-6*exp(2^(1/2))+2^(1/2)*exp(2^(1/2))^2-
2^(1/2))/exp(2^(1/2))*cosh(2^(1/2)*x)
```

显然所显示的结果很不直观，利用 pretty 命令将结果化为手写格式：

```
pretty(y)
```

以上命令的执行结果为：

```
                 1/2   4   1/2             1/2   3             1/2       1/2          1/2
           (exp(2    )   2      - 6 exp(2    )  + 6 exp(2     ) + 2     ) sinh(2      x)
  x + 1/2 ----------------------------------------------------------------
                                              1/2   2              1/2
                                        (exp(2    )   + 1) exp(2     )

                        1/2          1/2        1/2   2      1/2          1/2
              (-6 exp(2      ) + 2       exp(2     )   - 2      ) cosh(2      x)
      - 1/2 ----------------------------------------------------
                                                  1/2
                                            exp(2     )
```

（2）求解常微分方程：$\frac{d^3y}{dx^3}-\frac{d^2y}{dx^2}=x$，且满足：$y(2)=5, y'(1)=2, y''(2)=4$。可以执行以下命令：

```
y=dsolve('D3y-D2y=x','y(2)=5','Dy(1)=2','D2y(2)=4','x')
```

以上命令的执行结果为：

```
y =
-1/2*x^2-1/6*x^3+1/3*(-17*exp(2)+42*exp(1))/exp(2)-7/2*(-exp(2)+2*exp(1))/exp(2)*x+7/exp(2)*exp(x)
```

（3）求解常微分方程：$w'''=-w$，初始条件为：$w(0)=1, w'(0)=0, w''(0)=0$。可以执行以下命令：

```
w=dsolve('D3w=-w','w(0)=1,Dw(0)=0,D2w(0)=0')
```

以上命令的执行结果为：

```
w =
```

```
1/3*exp(-t)+2/3*exp(1/2*t)*cos(1/2*3^(1/2)*t)
```

若想使得到的结果 $w$ 为自变量 $x$ 的表达式，只需执行以下命令：

```
w=dsolve('D3w=-w','w(0)=1,Dw(0)=0,D2w(0)=0','x')
```

以上命令的执行结果为：

```
w =
1/3*exp(-x)+2/3*exp(1/2*x)*cos(1/2*3^(1/2)*x)
```

### 3. 求解常微分方程组

使用 dsolve 命令不但可以求常微分方程组的通解，而且还可以求常微分方程组的特解，此时其调用格式为：

- dsolve('equ1', 'equ2',…, 'equn','var')
- dsolve('equ1, equ2,…, equn','var')
- dsolve('equ1', 'equ2',…, 'equn','condition1','condition2',…,'conditionn','var')
- dsolve('equ1, equ2,…, equn','condition1, condition2,…, conditionn','var')

前两种调用格式用于求解常微分方程组的通解，其功能完全一样。后两种调用格式用于求解常微分方程组的特解，其功能完全相同。

【例 5-28】

求微分方程组 $\begin{cases} f''=f+3g+\sin x \\ g'=f'+4+\cos x \end{cases}$ 的通解以及在初始条件：$f'(2)=0, f'(3)=3, g(5)=1$ 下的特解。可心执行以下命令：

```
[gen_f,gen_g]=dsolve('D2f=f+3*g+sin(x),Dg=Df+4+cos(x)','x')
[f,g]=dsolve('D2f=f+3*g+sin(x),Dg=Df+4+cos(x)',...
    'Df(2)=0,Df(3)=3,g(5)=1','x');
```

其中：...为续行符号。

以上命令的执行结果为：

```
gen_f =
3/4*C1+1/8*C1*exp(2*x)+1/8*C1*exp(-2*x)+1/4*C2*exp(2*x)-1/4*C2*exp(-2*x)+3/8*C3*exp(2*x)-3/4*
C3+3/8*C3*exp(-2*x)-3*x-4/5*sin(x)
gen_g =
1/8*C1*exp(2*x)-1/4*C1+1/8*C1*exp(-2*x)+1/4*C2*exp(2*x)-1/4*C2*exp(-2*x)+3/8*C3*exp(-2*x)+
3/8*C3*exp(2*x)+1/4*C3+x+1/5*sin(x)
```

特解 $f,g$ 的表达式都很复杂，这里略去，请读者自己上机运行。

### 4. 求解线性齐次常微分方程组

求解线性齐次常微分方程组 $X'=AX$ 的过程如下：若矩阵 $A$ 有 $n$ 个线性无关的特征矢量 $v1,v2,\cdots,vn$，它们对应的特征值分别为 $\lambda 1,\lambda 2,\cdots,\lambda n$（$\lambda i$ 与 $\lambda j$ 可以相同），则矩阵 $\varphi(t)=\left[e^{\lambda_1 t},e^{\lambda_2 t},\cdots,e^{\lambda_n t}\right]$，$t\in(-\infty,+\infty)$就是线性齐次常微分方程组 $X'=AX$ 的解。

上述求解过程用 MATLAB 语言可以很方便地实现，可以编制名为 dsolv.m 的函数文件，然后通过调用 dsolv 命令就可以很方便地求解线性齐次常微分方程组了。dsolv.m 函数的程序清单如下：

```
function y=dsolv(A)      %该函数可用来求线性齐次常微分方程组X'=AX的解，返回给y
syms t real              %符号变量说明
e=eig(A);                %求A的特征矢量
[v,d]=eig(A);            %求A的特征值矩阵d和特征值矢量矩阵v
y=exp(d*t)'*v;           %求得线性齐次常微分方程组X'=AX的解
```

【例 5-29】

求线性齐次常微分方程组 $X'=\begin{bmatrix}1 & 3\\ -3 & 4\end{bmatrix}X$ 的解。可以执行以下命令：

```
A=[1 3;-3 4];
X=dsolv(A)
```

以上命令的执行结果为：

```
X =
[ exp(t*(5/2-3/2*i*3^(1/2)))*(1/4*2^(1/2)-1/4*i*6^(1/2))+1/2*2^(1/2), exp(t*(5/2-3/2*i*3^(1/2)))* (1/4*2^
(1/2)+1/4*i*6^(1/2))+1/2*2^(1/2)]
[    1/4*2^(1/2)-1/4*i*6^(1/2)+1/2*exp(t*(5/2+3/2*i*3^(1/2)))*2^(1/2),     1/4*2^(1/2)+1/4*i*6^ (1/2)+1/2*
exp (t*(5/2+3/2*i*3^(1/2)))*2^(1/2)]
```

用 MATLAB 求线性非齐次常微分方程组的解也是十分方便的，与求解线性齐次常微分方程组相比，求解线性非齐次常微分方程组只是多了一个积分过程罢了。有兴趣的读者可以自己编写 MATLAB 函数完成上述功能。

## 5.5 反函数和复合函数的求法

利用 MATLAB 提供的 finverse 命令和 compose 命令可以方便地求解已知函数的反函数及其复合函数，本节将分别予以介绍。

### 5.5.1 求反函数（finverse）

MATLAB 提供了求解已知函数的反函数的命令：finverse，其调用格式有：

- *g*=finverse(f)：返回值 *g* 是已知函数 *f* 的反函数（自变量由系统的默认变量确定）。若 *f* 为单符号变量（如 *x*）的一个标量，那么 *g* 也是一个涉及符号的标量，并且满足 $g(f(x))=x$。
- *g*=finverse(f,v)：同上，但指定变量为 *v*。该命令在 *f* 含有多个符号的情况下，求它关于某个变量的反函数时要比第 1 种调用格式显的简洁明了。

【例 5-30】

（1）求函数 $f(x)=\dfrac{1}{\tan x}$ 的反函数。可以执行以下命令：

```
syms x
y=finverse(1/tan(x))
```

以上命令的执行结果为：

```
y =
atan(1/x)
```

（2）已知$f=x^2+y$，求$f$关于变量$y$的反函数，$f$关于变量$x$的反函数。可以执行以下命令：

```
syms x y                %符号变量说明
f=x^2+y;                %创建符号表达式
f1=finverse(f,y)        %求 f 关于变量 y 的反函数 f1
f2=finverse(f)          %求 f 关于变量 x 的反函数 f2
```

以上命令的执行结果为：

```
f1 =
-x^2+y
Warning: finverse(x^2+y) is not unique.
> In D:\MATLAB6P1\toolbox\symbolic\@sym\finverse.m at line 43
f2 =
(-y+x)^(1/2)
```

计算过程中给出警告信息：$f$关于变量$x$的反函数不是惟一的，结果只给出正的平方根形式。

### 5.5.2 求复合函数（compose）

利用 MATLAB 提供的 compose 命令可以方便地求解复合函数，其调用格式包括：

- compose(f,g)：返回值为$f(g(y))$，其中$f=f(x),g=g(y)$。这里的$x$、$y$分别是由 findsym 函数确定的$f$、$g$中的符号变量。
- compose(f,g,z)：返回值为$f(g(z))$，其中$f=f(x),g=g(y)$且$x$、$y$分别是由 findsym 函数确定的$f$、$g$中的符号变量。
- compose(f,g,x,z)：返回值为$f(g(z))$且视$x$为$f$的自变量，即如果$f=\cos\left(\frac{x}{t}\right)$，则 compose(f,g,x,z)返回值就是$\cos\left(\frac{g(z)}{t}\right)$，而 compose(f,g,t,z)则返回$\cos\left(\frac{x}{g(z)}\right)$。
- compose(f,g,x,y,z)：返回值为$f(g(z))$且视$x$为$f$的自变量，视$y$为$g$的自变量。例如：$f=\cos\left(\frac{x}{t}\right)$，$g=\sin\left(\frac{y}{u}\right)$则 compose(f,g,x,y,z)返回值为$\cos\left(\frac{\sin\left(\frac{z}{u}\right)}{t}\right)$，而 compose (f,g,x,u,z)则返回为$\cos\left(\frac{\sin\left(\frac{y}{z}\right)}{t}\right)$。

【例 5-31】

已知函数 $f=\frac{1}{1+x^2}$， $g=\sin(y)$， $h=x^t$ 和 $p=e^{-\frac{y}{u}}$。可以执行以下命令：

```
syms x y z t u
f=1/(1+x^2);
g=sin(y);
h=x^t;
p=exp(-y/u);
fg=compose(f,g)
```

以上命令的执行结果为：

```
fg =
1/(sin(y)^2+1)
```

若执行命令：

```
fgt=compose(f,g,t)
```

则显示结果为：

```
fgt =
1/(sin(t)^2+1)
```

若执行命令：

```
hg=compose(h,g,x,z)
```

则显示结果为：

```
hg =
sin(z)^t
```

若执行命令：

```
hgt=compose(h,g,t,z)
```

则显示结果为：

```
hgt =
x^sin(z)
```

若执行命令：

```
hgz=compose(h,p,x,y,z)
```

则显示结果为：

```
hgz =
exp(-z/u)^t
```

若执行命令：

```
hpz=compose(h,p,t,u,z)
```

则显示结果为：

```
hpz =
x^exp(-y/z)
```

由上面的例子可以看出，利用 MATLAB 求复合函数是非常方便的。希望读者认真练习并加以掌握。

## 5.6 小结

本章主要介绍了 MATLAB 符号运算的具体实现途径。讲述了 MATLAB 在微积分、极限、级数、符号方程（组）中的实际应用情况，并且讲述了如何用 MATLAB 求解反函数和复合函数等内容。可以说，用 MATLAB 几乎可以解决常见的一切数学问题。希望读者能熟练掌握本章所讲述的所有内容。

# 第6章　高级符号运算功能的实现

**知识点：**

- 积分变换
- 几个补充命令和特殊函数
- 可控精度的实现以及抽象函数的创建
- 如何在 MATLAB 中使用 MAPLE
- 函数计算器及泰勒计算器的使用

本章介绍了 MATLAB 在实现积分变换方面所提供的有关命令，并介绍了 double、poly2sym 等补充命令的调用方法。借助于 MATLAB 提供的特殊函数命令，读者可以方便地实现各种特殊数学函数。另外，MATLAB 可以根据用户的需要方便地设置数据的精度，并可以方便地创建抽象函数。本章还介绍了如何在 MATLAB 中使用 MAPLE，此部分内容是 MATLAB 符号运算功能的一个延伸。本章的最后则介绍了 MATLAB 所提供的两个计算器：函数计算器及泰勒计算器。读者会发现灵活使用这两个计算器往往能收到事半功倍的效果。

## 6.1　积分变换

MATLAB 提供了丰富的积分变换命令，当读者掌握了这些命令以后就会发现使用 MATLAB 实现复杂的积分变换是很容易的一件事情。本节将介绍 MATLAB 所提供的积分变换命令。

### 6.1.1　傅立叶变换及其逆变换

对函数 $f(x)$ 进行傅立叶（fourier）变换：$f=f(x)\Rightarrow F=F(w)$ 的计算公式为：$F(w)=\int_{-\infty}^{+\infty}f(x)e^{-iwx}\mathrm{d}x$

MATLAB 提供了对函数进行傅立叶变换的命令 fourier，其调用格式有：

- F=fourier (f)：对自变量为 $x$ 的表达式 $f(x)$ 进行变换，返回值为 $F(w)$；当 $f$ 是变量 $w$ 的函数即 $f=f(w)$ 时，则变换结果为 $F=F(t)$；如果表达式中没有 $t$ 和 $x$ 这两个变量，则该命令将对系统默认的变量进行傅立叶变换。
- F=fourier (f,v)：指定了变换结果为变量 $v$ 的函数。表现在计算过程中则公式变为：$F(v)=\int_{-\infty}^{+\infty}f(x)\mathrm{e}^{-ivx}\mathrm{d}x$。用 MATLAB 语言则可描述为：fourier(f,v)$\Leftrightarrow F(v)$ =int(f(x)*exp(-i*v*u),u,-inf,inf)。
- F=fourier(f,u,v)：指定了要对函数表达式作关于变量 $u$ 的 fourier 变换。对应此时的傅

立叶变换公式变为： $F(v)=\int_{-\infty}^{+\infty}f(u)e^{-ivu}\mathrm{d}u$ 。

对函数 $F(x)$ 进行傅立叶逆变换： $F=F(w)\Rightarrow f=f(x)$ 的计算公式为： $f(x)=\frac{1}{2\pi}\int_{-\infty}^{+\infty}F(w)e^{iwx}\mathrm{d}w$ 。

MATLAB 实现傅立叶逆变换的命令为 ifourier，其调用格式有：

- f=ifourier (F)：$F$ 为待进行傅立叶逆变换的代数表达式，该命令对 $F(w)$实行傅立叶逆变换得到一个自变量为 $x$ 的函数 $f(x)$。如果 $F=F(x)$则该命令将返回一个自变量为 $t$ 的函数 f (t)。
- f=ifourier (F,u)
- f=ifourier (F,v,u)

后两种调用格式中 $u,v$ 的用法和 fourier 命令中的用法完全一致，如 f=ifourier(F,v,u)表示对表达式 $F$ 进行关于自变量为 $v$ 的傅立叶逆变换，并将结果表示为变量 $u$ 的函数。用公式可表示为： $f(u)=\frac{1}{2\pi}\int_{-\infty}^{+\infty}F(v)e^{ivu}\mathrm{d}v$ ，用 MATLAB 语言可以表示为：ifourier(F,v,u)⇔f(u)=1/(2*pi)*int(F(v)*exp(I*v*u),v,-inf,inf)。

【例 6-1】

（1）对函数 $f(x)=e^{-x^2}$ 进行傅立叶变换，可以执行以下命令：

```
syms x
F=fourier(exp(-x^2))
```

以上命令的执行结果为：

```
F =
pi^(1/2)*exp(-1/4*w^2)
```

（2）求函数 $f(t)=\frac{1}{t}$ 的傅立叶变换表达式，可以执行以下命令：

```
syms t
F=fourier(1/t)
```

以上命令的执行结果为：

```
F=
i*pi*(Heaviside(-w)-Heaviside(w))
```

（3）求函数 $F(x)$导数的傅立叶变换表达式，可以执行以下命令：

```
syms x w
fourier(diff(sym('F(x)')),x,w)
```

则得到的执行结果为：

```
ans =
i*w*fourier(F(x),x,w)
```

（4）试求函数 $f(x)=xe^{-|x|}$ 的傅立叶变换结果。可以执行以下命令：

```
syms x
F=fourier(x*exp(-abs(x)))
```

以上命令的执行结果为：

```
F =
-4*i/(1+w^2)^2*w
```

【例 6-2】

（1）试对函数 $F(a)=e^{-\frac{a^2}{4}}$ 进行逆傅立叶变换，并把所得结果表示为自变量为 $b$ 的函数。可以执行以下命令：

```
syms a b
f=ifourier(exp(-a^2/4),a,b)
```

以上命令的执行结果为：

```
f =
1/2*4^(1/2)/pi^(1/2)*exp(-b^2)
```

（2）执行以下命令：

```
syms t u v w x
f1=ifourier(w*exp(-3*w)*sym('Heaviside(w)'))
%求 exp(-3*w)*Heaviside(w)的傅立叶逆变换
f2=ifourier(1/(1+w^2),u)
%求 1/(1+w^2)的傅立叶逆变换，并把结果表示为 u 的函数
f3=ifourier(v/(1+w^2),v,u)
%v/(1+w^2)的傅立叶逆变换，并把结果表示为 u 的函数
f4=ifourier(sym('fourier(f(x),x,w)'),w,x)
%对 f(x)先进行傅立叶变换在进行傅立叶逆变换则返回函数本身
```

则执行结果为：

```
f1 =
1/2/(-3+i*x)^2/pi
f2 =
1/2*exp(-u)*Heaviside(u)+1/2*exp(u)*Heaviside(-u)
f3 =
-i/(1+w^2)*Dirac(1,u)
f4 =
f(x)
```

### 6.1.2 拉普拉斯变换及其逆变换

对函数 $f(x)$ 进行拉普拉斯（laplace）变换：$F=F(t)\Rightarrow L=L(s)$ 的计算公式为：

$$L(s)=\int_0^{+\infty}F(t)e^{-st}\mathrm{d}t\text{ 。}$$

MATLAB 实现拉普拉斯变换的命令为 laplace，其调用格式有：

- L=laplace (F)：$F$ 为待进行拉普拉斯变换的代数表达式，其默认变量为 $t$，若 $F$ 不含 $t$ 则针对系统默认的变量对表达式进行拉普拉斯变换，该命令返回的函数其默认自变量为 $s$。如果 $F=F(s)$那么该命令的返回结果为 $L=L(t)$。该命令可用 MATLAB 语言描述为：laplace(F) ⇔L(s)=int(F(t)*exp(-s*t),0,inf)并且积分针对变量 $t$ 进行。
- L=laplace (F,t)：指定返回结果 $L$ 为自变量为 $t$ 的函数，而不是系统默认的 $s$，用 MATLAB 语言可以描述为：laplace(F,t) ⇔L(t)=int(F(x)*exp(-x*t),0,inf)。用公式可表示为：$L(t)=\int_0^{+\infty}F(x)e^{-tx}\mathrm{d}x$。
- L=laplace(F,w,z)：要求 $F$ 对变量 $w$ 进行拉普拉斯变换，返回值的自变量指定为 $z$，而不是系统默认的 $s$。用 MATLAB 语言以描述为：laplace(F,w,z) ⇔L(z)=int(F(w)*exp(-z*w),0,inf)。用公式可表示为：$L(z)=\int_0^{+\infty}F(w)e^{-zw}\mathrm{d}w$。

对函数 $L(s)$ 进行拉普拉斯逆变换的即 $L=L(s)\Rightarrow F=F(t)$ 的计算公式为：$F(t)=\int_{c-i\infty}^{c+i\infty}L(s)e^{st}\mathrm{d}s$。

MATLAB 实现拉普拉斯逆变换的命令为 ilaplace，其调用格式有：

- F=ilaplace(L)：$L$ 为待进行拉普拉斯逆变换的表达式，其默认变量为 $s$，该命令返回值 $F$ 的默认自变量为 $t$；如果 $L=L(t)$那么该命令的返回结果为 $F=F(x)$；该命令可用 MATLAB 语言描述为：F(t)=int(L(s)*exp(s*t),s,c-i*inf,c+i*inf)，其中 $c$ 是选择的一个实数，它能确保 $L(s)$的所有奇点都位于直线 $s=c$ 的左边。上面的 int 命令针对变量 $s$ 进行，即 $F(t)=\int_{c-i\infty}^{c+i\infty}L(s)e^{st}\mathrm{d}s$。
- F=ilaplace(L,y)：指定返回值 $F$ 的变量为 $y$ 而不是系统默认的 $t$。积分仍针对 $L$ 中的变量 $s$ 进行，用 MATLAB 语言可描述为：ilaplace (L,y)⇔$F(y)$=int (L(y)*exp(s*y),s,c-i*inf,c+i*inf)。用公式可描述为：$F(y)=\int_{c-i\infty}^{c+i\infty}L(y)e^{sy}\mathrm{d}s$。
- F=ilaplace(L,y,x)：针对变量 $y$ 求 $L$ 的逆拉普拉斯变换，其结果为 $x$ 的函数。用 MATLAB 语言可描述为：ilaplace(L,y,x) ⇔F(y)=int(L(y)*exp(x*y),y,c-i*inf,c+i*inf)。用公式可描述为：$F(x)=\int_{c-i\infty}^{c+i\infty}L(y)e^{xy}\mathrm{d}y$。

【例 6-3】

（1）执行以下命令：

```
syms a s t w x
L1=laplace(t^5)                    %对 t^5 进行拉普拉斯变换
L2=laplace(exp(a*s))               %对 exp(a*s)进行拉普拉斯变换
L3=laplace(sin(w*x),t)             %对 sin(w*x)进行拉普拉斯变换并把结果表示为 t 的函数
L4=laplace(cos(x*w),w,t)           %对 cos(w*x) 进行拉普拉斯变换且使 L4 成为 t 的函数
L5=laplace(diff(sym('F(t)')))      %先求 F(t) 的微分再求拉普拉斯变换
```

以上命令的执行结果为：

```
L1 =
120/s^6
L2 =
1/(t-a)
L3 =
w/(t^2+w^2)
L4 =
t/(t^2+x^2)
L5 =
s*laplace(F(t),t,s)-F(0)
```

（2）试对函数 $F(t)=\sin(xt+2t)$进行拉普拉斯变换，并试把所得结果表示为变量 $v$ 的函数。可以执行以下命令：

```
syms t v x s
F=sin(x*t+2*t);
L1=laplace(F)          %L1 为变量 x 的函数
L2=laplace(F,s,v)           %L2 为变量 v 的函数
以上命令的执行结果为：
L1 =
(x+2)/(s^2+(x+2)^2)
L2 =
sin(x*t+2*t)/v
```

（3）试求下列函数的逆拉普拉斯变换：$L_1=\frac{1}{s-1}$、$L_2=\frac{1}{t^2+1}$、$L_3=t^{-\frac{5}{2}}$以及 $L_4=\frac{y}{y^2+w^2}$，并把 $L_3$、$L_4$ 表示为变量 $x$ 的函数。可以执行以下命令：

```
syms s t w x y
F1=ilaplace(1/(s-1))                 %所得结果是变量 t 的函数
F2=ilaplace(1/(t^2+1))               %所得结果是变量 x 的函数
F3=ilaplace(t^(-5/2),x)              %将所得结果表示为变量 x 的函数
F4=ilaplace(y/(y^2+w^2),y,x)         %针对变量 y 进行逆拉普拉斯变换并将所得结果表
                                     %示为变量 x 的函数
```

以上命令的执行结果为：

```
F1 =
exp(t)
F2 =
sin(x)
F3 =
4/3*x^(3/2)/pi^(1/2)
F4 =
cos((w^2)^(1/2)*x)
```

（4）显然对一个函数先进行拉普拉斯变换，再求其逆变换则返回该函数本身，执行以

下命令：

```
syms s x
F=ilaplace(sym('laplace(F(x),x,s)'),s,x)  %先对 F(x)进行拉普拉斯变换，再求其逆变换
                                          %则返回 F(x)本身
```

以上命令的执行结果为：

```
F =
F(x)
```

### 6.1.3 Z变换及其逆变换

对函数 $f(x)$进行 Z 变换即 $f=f(n) \Rightarrow F=F(z)$的计算公式为：$F(z)=\sum_{n=0}^{\infty}\frac{f(n)}{z^n}$。MATLAB 实现 Z 变换的命令为 ztrans，其调用格式有：

- F=ztrans(f)：$f$ 为待进行 Z 变换的表达式；$f$ 的默认变量为 $n$，返回值是变量 $z$ 的函数：$F=F(z)$。若 $f$ 的自变量为 $z$ 则该命令的返回值是变量 $w$ 的函数：$F=F(w)$；若 $f$ 中没有 $n,z$ 这两个变量，那么该命令针对系统默认的变量进行变换。该命令用 MATLAB 语言可以描述为 $F(w)$=symsun(f(n)/w^n,n,0,inf)
- F=ztrans(f,w)：同上，只不过指定返回值是变量 $w$ 的函数（而不是系统默认的 $z$），用 MATLAB 语言可以描述为：ztrans(f,w) ⇔F(w)=symsum(f(n)/w^n,n,0,inf)。
- F=ztrans(f,k,w)：针对 $f$ 中的变量 $k$ 进行 $z$ 变换，返回值是变量为 $w$ 的函数。用 MATLAB 语言可以描述为：ztrans(f,k,w) ⇔F(w)=symsum(f(k)/w^k,k,0,inf)。

对函数 $F(z)$实现 Z 逆变换即 $F=F(z) \Rightarrow f=f(n)$的计算公式为：$f(n)=\frac{1}{2\pi i}\oint_{|z|=R}F(z)z^{n-1}\mathrm{d}z$，其中 $n$=1,2,…。MATLAB 实现 Z 逆变换的命令为：iztrans，其调用格式有：

- f=iztrans(F)：式中 $F$ 为待进行 Z 逆变换的表达式，$F$ 的默认变量为 $z$。该命令的返回值是变量为 $n$ 的函数：$f=f(n)$；若 $F$ 的自变量是 $n$，则该命令的返回值是变量为 $k$ 的函数：$f=f(k)$；如果 $F$ 中不含变量 $z$、$n$，则该命令将针对系统的默认变量进行变换。
- $f$=iztrans(F,k)：同上，只不过返回值是自变量为 $k$ 的函数。
- $f$=iztrans(F,w,k)：针对变量 $w$ 求 $F$ 的 Z 逆变换，且返回值上是变量为 $k$ 的函数。

【例 6-4】

（1）针对下列函数求其 Z 变换：$f(n)=2^n$、$g(n)=\sin(kn)$、$h(k)=\cos(kn)$和 $h(n)=\cos(kn)$。要求：把 $g(n)$的返回结果表示成变量 $w$ 的函数；把 $h(k)$的返回结果表示成变量 $z$ 的函数；把 $h(n)$的返回结果表示成变量 $w$ 的函数。可以执行以下命令：

```
syms k n w z                   %符号变量说明
F=ztrans(2^n)                  %求函数 2^n 的 Z 变换
G=ztrans(sin(k*n),w)           %求函数 sin(kn)的 Z 变换并把结果表示成 w 的函数
HK=ztrans(cos(k*n),k,z)        %求函数 cos(kn)关于变量 k 的 Z 变换，并把结果表示
                               %成 z 的函数
```

```
HN=ztrans(cos(k*n),n,w)        %求函数 cos(kn)关于变量 n 的 Z 变换，把结果表示成
                               %w 的函数
```

以上命令的执行结果为：

```
F =
1/2*z/(1/2*z-1)
G =
w*sin(k)/(w^2-2*w*cos(k)+1)
HK =
(z-cos(n))*z/(z^2-2*z*cos(n)+1)
HN =
(w-cos(k))*w/(1+w^2-2*w*cos(k))
```

如求函数 $f(n+1)$的 Z 变换，执行以下命令：

```
H=ztrans(sym('f(n+1)'))     %求函数 f(n+1)的 Z 变换
```

则执行结果为：

```
H =
z*ztrans(f(n),n,z)-f(0)*z
```

（2）已知函数 $f$=sin($xt$+2$t$)，试求 $f$ 分别关于变量 $x,t$ 的 $Z$ 变换。要求把 $f$ 关于 $t$ 的 $Z$ 变换后所得的结果表达为变量 $y$ 的函数。可以执行以下命令：

```
syms   x y t                %符号变量说明
f=sin(x*t+2*t);
f1=ztrans(f)
f2=ztrans(f,t,y)
以上命令的执行结果为：
f1 =
(2*z*cos(t)-1)*z*sin(t)/(1+z^2-2*z*cos(t))
f2 =
(-y^2*sin(x)+2*y^2*sin(x)*cos(1)^2+2*y^2*cos(x)*cos(1)*sin(1)-8*y*cos(1)^3*sin(1)+4*y*cos(1)*sin(1)-
2*y*cos(x)*sin(x)-sin(x)+2*sin(x)*cos(1)^2+2*cos(x)*cos(1)*sin(1))*y/(16*y^2*cos(1)^4-8*y*cos(x)*cos(1)^2-
8*y^3*cos(x)*cos(1)^2-16*y^2*cos(1)^2+1+2*y^2+y^4+4*y*cos(x)+4*y^3*cos(x)+4*y^2*cos(x)^2)
```

（3）求函数 $F(z)=\dfrac{z}{z-2}$、$G(z)=e^{\frac{x}{z}}$、$H(n)=\dfrac{n(n+1)}{n^2+2n+1}$的 $Z$ 逆变换结果。要求：$G(z)$对应的结果是变量为 k 的函数。可以执行以下命令：

```
syms   x z k n                     %符号变量说明
f=iztrans(z/(z-2))                 %求 F(z)的 Z 逆变换
gk=iztrans(exp(x/z),z,k)           %求 G(z)对应的 Z 逆变换，返回结果是变量 k 的
                                   %函数
h=iztrans(n*(n+1)/(n^2+2*n+1))     %求 H(n)的 Z 逆变换
```

以上命令的执行结果为：

```
f =
2^n
gk =
x^k/k!
h =
(-1)^k
```

### 6.1.4 信号处理中常用的数值变换命令

在本小节中将介绍信号处理中常用的几个数值变换命令。

线性调频 Z 变换命令（czt）

其调用格式为：

- y=czt(x)：进行信号 $x$ 的线性调频 Z 变换。
- y=czt(x,m,w,a)：同上，参数 $m$ 用来设定长度；参数 $w$、$a$ 则用来定义 $z$ 变换所沿的螺旋周线。

#### 1．离散余弦变换及其逆变换命令（dct/idct）

离散余弦变换命令 dct 的调用格式为：

- y=dct(x)：对信号矢量 $x$ 进行离散余弦变换，返回矢量 $y$。$y$ 的大小与 $x$ 相等，其元素就是离散余弦变换的系数。
- Y=dct(x,n)：当信号矢量 $x$ 的长度不等于 $n$ 时，通过补零或者截断的方法来整理 $x$，使之长度为 $n$，然后对 $x$ 进行离散余弦变换并把结果返回给 $y$。

以上命令中，若 $x$ 是矩阵，则变换针对 $x$ 的每一列进行。

离散余弦变换逆变换命令 idct 的调用格式有：

- x=idct(Y)：如果 $y$ 是由 dct(x)得来的，那么执行 x=idct(y)则返回原始的信号矢量 $x$。
- x=idct(Y,n)：同上，但在变换前先对矢量 $y$ 进行补零或者截断来整理 $y$，使之长度为 $n$。同样，若 $y$ 是个矩阵则 idct 命令针对 $y$ 的每列进行运算。

#### 2．快速傅立叶变换及其逆变换

MATLAB 既可以进行一维、二维的快速傅立叶（逆）变换，又可以进行 $n$ 维的快速傅立叶（逆）变换，下面分别加以论述。

（1）一维快速傅立叶变换及其逆变换。MATLAB 提供的 fft 命令可以完成一维快速傅立叶变换，而 ifft 命令则可以完成一维快速傅立叶逆变换。

fft 命令的调用格式有：

- fft(x)：对矢量 $x$ 进行离散傅立叶变换。如 $x$ 的长度是 2 的正整数次幂，则该命令采用快速傅立叶变换：FFT。应当注意的是，这样的变换是没有经过规格化的。
- fft(x,n)：返回一个长度为 $n$ 的矢量，其元素是矢量 $x$ 中前 $n$ 个元素的离散傅立叶变换值，如果 $x$ 的元素个数不足 $n$，则通过在 $x$ 后面补零的办法补齐。
- fft(A)：按矩阵 $A$ 的列进行离散傅立叶变换，返回值为一矩阵。
- fft(A,n,dim)或 fft(A,[],dim)：返回多维数组 $A$ 中 dim 维内的列离散傅立叶变换阵。

ifft 命令的调用格式有：

- ifft(x)
- ifft(x,n)
- ifft(A)
- ifft(A,n,dim) 或 ifft(A,[],dim)

以上调用格式中，各个参数的意义和 fft 命令中对应参数的意义完全相同，ifft 命令和 fft 命令是互逆的。事实上，对长度为 $N$ 的矢量 $x$ 来说，fft 命令是按照下式进行计算的：

$$X(k)=\sum_{n=1}^{N}x(n)e^{-2j\pi\frac{(k-1)(n-1)}{N}}$$，其中 $1\leqslant k\leqslant N$。

ifft 命令是按照下式进行计算的：

$$x(n)=\frac{1}{N}\sum_{k=1}^{N}X(k)e^{2j\pi\frac{(k-1)(n-1)}{N}}$$，其中 $1\leqslant n\leqslant N$。

【例 6-5】

执行以下命令：

```
x=linspace(0,1);                              %把[0,1]区间分成间距相等的 100 等份，得到
                                              %矢量 x
y=[linspace(0,1,50) linspace(1,0,50)];        %得到矢量 y
subplot(1,3,1)                                %创建子图窗口
plot(x,y)                                     %绘制 x,y 构成的曲线
title('锯齿形函数')                            %进行图形标注
subplot(1,3,2)
plot(x,fft(y))                                %绘制矢量 x 与 y 经傅立叶变换后的矢量所构
                                              %成的曲线
title('傅立叶变换函数')
subplot(1,3,3)
plot(x,ifft(fft(y)))                          %绘制矢量 x 与 y 经过傅立叶变换后进行逆傅
                                              %立叶变换的矢量所构成的曲线
title('求傅立叶变换的逆变换')
```

以上命令的执行结果为：

```
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
```

上面程序所绘制出的图形如图 6-1 所示。

由执行过程给出的提示可知：在进行快速傅立叶变换时，fft 和 ifft 命令将把矢量的虚部忽略。

（2）二维快速傅立叶变换及其逆变换。MATLAB 提供的 fft2 命令能对信号矩阵进行二维快速傅立叶变换，ifft2 命令则可完成二维快速傅立叶逆变换。

- fft2(A)：对矩阵 $A$ 进行二维离散傅立叶变换，该命令的返回结果为一矩阵。要注意的是得到的结果矩阵并没有进行正交化处理；如果 $A$ 是个矢量，那么 fft2(A)与 fft(A)功能完全相同。

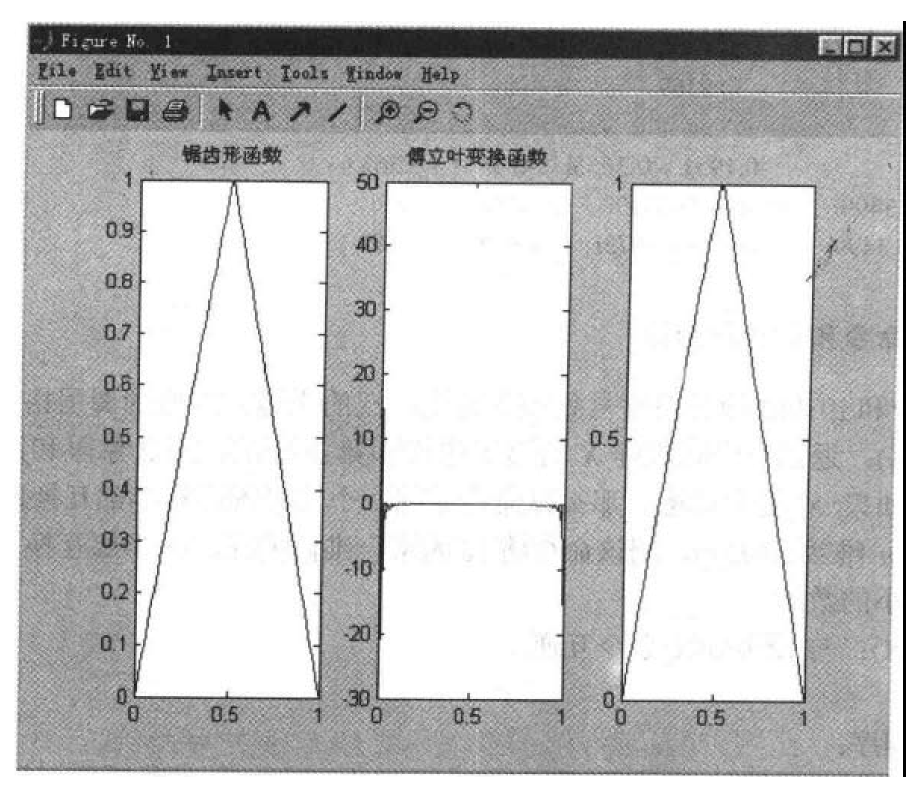



图 6-1　傅立叶变换所得的曲线

● fft2(A,m.n)：同上，但在变换前先把 $A$ 通过补零或者截断的方法处理成 $m\times n$ 的矩阵。

● ifft2(A)：对矩阵 $A$ 进行二维离散傅立叶逆变换，所得结果也是一个矩阵。

● ifft2(A,m,n)：同上，$m$、$n$ 的意义和 fft2 (A)中的意义相同。

事实上，fft2(A)命令的计算过程是：先对 $A$ 的每一列进行一维快速傅立叶变换，然后对得到的中间结果（行矢量）再进行一次一维快速傅立叶变换，从而得到二维快速傅立叶变换。

（3）多维快速傅立叶变换及其逆变换。fftn 命令、ifftn 命令分别可以进行多维快速傅立叶变换和其逆变换的运算。它们的调用格式如下：

● fftn(A,size)：进行 $n$ 维数组 $A$ 的 $n$ 维离散傅立叶变换，返回结果为一数组。如果 $A$ 是一个矢量，则得到的结果也是一个矢量。如果给定了数组的大小 size，则在执行变换之前先把 $A$ 重构成 size 大小。

● ifftn(A,size)：求 $n$ 维数组 $A$ 的 $n$ 维离散傅立叶逆变换，返回结果为一数组。如果 $A$ 是一个矢量，则得到的结果也是一个矢量。如果给定了数组的大小 size，则在执行变换之前先把 $A$ 重构成 size 大小。

【例 6-6】

执行以下程序：

```
A=rand(3)              %A 为 3*3 的随机矩阵
B=fftn(A,[3,3])        %对 A 进行离散傅立叶变换
```

显示结果为：

```
A =
    0.4447    0.9218    0.4057
```

```
    0.6154    0.7382    0.9355
    0.7919    0.1763    0.9169
B =
   5.9464                -0.1951 + 0.3653i  -0.1951 - 0.3653i
  -0.3149 - 0.3499i  -0.6386 - 0.4489i   0.1765 + 1.2573i
  -0.3149 + 0.3499i   0.1765 - 1.2573i  -0.6386 + 0.4489i
```

### 3．fftshift 命令和 iffshift 命令

fftshift 命令和 iffshift 命令在信号处理中有着广泛的用途，它们的调用格式为：

- fftshift(X)：返回一个将矩阵 *X* 的第 1 象限和第 3 象限、第 2 象限和第 4 象限互换的数组；如果 *X* 是个矢量，那么该命令返回一个左半部和右半部互换的矢量；如果 *X* 是一个 *n* 维数组(*n*>2)，则该命令将 *X* 的每一维内的左、右半部互换，返回一个与 *X* 同样大小的数组。
- ifftshift(X)：与 fftshift(X)命令互逆。

【例 6-7】

执行以下程序：

```
X=[1 2 3 4 5];          %对矢量 X 进行赋值
xshft=fftshift(X)        %由于 X 为矢量，因此返回结果为 X 的左、右半部互换后所得的矢量
A=rand(3)                %A 为 3*3 的随机矩阵
B=fftshift(A)
C=fftn(A)                %对 A 进行离散傅立叶变换
D=fftshift(B)
```

显示结果为：

```
xshft =
    4     5     1     2     3
A =
    0.4103    0.3529    0.1389
    0.8936    0.8132    0.2028
    0.0579    0.0099    0.1987
B =
    0.1987    0.0579    0.0099
    0.1389    0.4103    0.3529
    0.2028    0.8936    0.8132
C =
     3.0781               0.5037 - 0.5504i    0.5037 + 0.5504i
    -0.1860 - 1.4230i    -0.6047 - 0.3770i    0.5942 - 0.3714i
    -0.1860 + 1.4230i     0.5942 + 0.3714i   -0.6047 + 0.3770i
D =
    0.8132    0.2028    0.8936
    0.0099    0.1987    0.0579
    0.3529    0.1389    0.4103
```

### 4．数字滤波命令

filter 命令可以实现一维滤波，filter2 命令则可以实现二维滤波操作，以下是命令的调用格式：

- y=filter(b,a,x)：用矢量 $a,b$ 描述所组成的滤波器对矢量 $x$ 进行滤波，返回经过滤波后的数据 $y$；如果 $x$ 是个矩阵，则针对 $x$ 的列进行操作；如果 $x$ 是个数组，则滤波沿第一个非单元素集合的维数进行。
- [y,zf]=filter(b,a,x,zi)：获得延滞的初始条件 zi 和终止条件 zf。其中 zi=length(max(length(a),length(b))-1)，或者对 $x$ 的每一列来说均满足类似条件的一个数组。
- filter(b,a,[],dim)或 filter(b,a,x,zi,dim)：沿着第 $m$ 维进行滤波操作。
- y=filter2(b,x)：利用矩阵 $b$ 中的 2 维 FIR 滤波器过滤 $x$ 中的数据。返回值 $y$ 是通过 2 维卷积计算得到的，并且包含卷积的中心部分，大小与 $x$ 相同。
- y=filter2(b,x,'shape')：返回值 $y$ 由 2 维卷积计算得出，其维数由参数 shape 决定，shape 可以是下列有效值之一：

  'same'：这也是缺省情况下的默认值，此时与命令 filter2(b,x)完全相同。

  'valid'：只返回卷积的一部分，该部分是通过不带 0 插值边缘计算得到的。此时，size(y) < size(x)。

  'full'：返回 2 维卷积的全部，此时 size(y) > size(x)。
- y=filtfilt(b,a,x)：功能与 filter(b,a,x)命令相似，只不过该命令实现的是零相位向前和向后数字滤波。要注意的是：$x$ 的长度由 max(length(b)-1,length(a)-1)决定并且必须大于滤波器阶数的 3 倍。另外，该命令不应当使用微分器以及希尔伯特 FIR 滤波器，因为这些滤波器的运算很大程度上依赖于它们的相位响应。

## 6.2　几个补充命令和特殊函数

### 6.2.1　几个补充命令

MATLAB 符号工具箱中给出了将矩阵转化为双精度类型矩阵的命令 double；由多项式系数矢量得到符号多项式表达式的命令 poly2sym；提取多项式系数的命令 sym2poly；创建字符型矩阵的命令 char；由 MATLAB 语言表达式获得 C 语言代码和获得 FORTRAN 语言代码的命令 ccode 及 fortran。了解这些命令，有时可以收到事半功倍的效果。下面就对上述命令进行简要介绍。

#### 1．double 命令

该命令的调用格式为：

double(X)：返回值是 $X$ 的双精度矩阵。如果 $X$ 已经是双精度类型的矩阵，那么执行该命令后矩阵 $X$ 不会发生什么变化。

通常在 for，if，while 等循环语句和判断语句中使用 double 命令，以使变量能够满足双精度类型的要求。

### 2. poly2sym 命令

若已经知道多项式的系数矢量，则可借助该命令得到对应该系数矢量的多项式（这里的“2”理解为“to”，poly2sym 即 polynome to symbol）。该命令有以下两种调用格式：

- poly2sym(C)：返回值为对应矢量 *C* 的多项式表达式，返回结果的默认变量是 *x*。
- poly2sym(C,'v')或 poly2sym(C,sym('v'))：指定所返回的多项式表达式变量为“*v*”，而不是系统默认的“*x*”。

【例 6-8】

（1）执行命令：

```
px=poly2sym([1 0 -2 -5])
```

显示结果为：

```
px =
x^3-2*x-5
```

由此可见，矢量的长度值与所得多项式的最高次数相差 1，这是因为矢量的最后一个元素对应于多项式常数项的原因。

（2）执行命令：

```
pt=poly2sym([1 0 -2 -5],'t')
```

显示结果为：

```
pt =
t^3-2*t-5
```

由此可见，poly2sym 命令的第 2 种调用格式可以指定多项式的变量。

而执行命令：

```
pt=poly2sym([1 0 -2 -5],sym('t'))
```

显示结果为：

```
pt =
t^3-2*t-5
```

由此可见，poly2sym(C,'v')与 poly2sym(C,sym('v'))命令的功能完全相同。

### 3. sym2poly 命令

该命令与 poly2sym 命令的功能正好相反。若已知多项式，则可通过该命令获得多项式系数。其调用格式为：

- sym2poly(P)：*P* 为多项式表达式，该命令的返回结果是一个行矢量，该矢量的元素是多项式的系数。

【例 6-9】

执行命令：

```
syms x t
v1=sym2poly(x^3-2*x-5)
v2=sym2poly(t^4-3*t^3+2*t^2-t+1)
```

显示结果为：

```
v1 =
     1     0    -2    -5
v2 =
     1    -3     2    -1     1
```

## 4．char 命令

该命令可用于创建字符型数组（字符串），其调用格式为：

- s=char(x)：其中数组 $x$ 中的正整数代表字符编码，该命令把 $x$ 转化为 MATLAB 字符数组（前 127 个编码为 ASCII 码）。显示的实际字符依赖于给定字体的编码字符。$x$ 中超出范围 0~65535 的任何元素，该命令将不予定义（该结论会因操作平台的不同而有所差别）；而使用 double 命令则可以将字符数组转化为它对应的数值编码形式。
- s=char(c)：当 $c$ 是字符串型的单元数组时，该命令将 $c$ 中的每个元素重新置到字符数组 s 的行上；而使用 cellstr 命令则可以重新转换回去。
- s=char(T1,T2,T3,⋯)：返回值为字符数组 $s$，$s$ 包含字符串 $T1,T2,T3,\cdots$ 等文本，并且将它们作为 $s$ 的行，每个字符串之间自动插入空格以获得有效矩阵；每个文本参数 $Ti$ 本身均可以是字符型矩阵。

## 5．ccode 命令

该命令能求得符号表达式的 C 语言源代码表达式，其调用格式为：

- ccode(s)：返回符号表达式 $s$ 的 C 语言编码形式。

【例 6-10】

（1）执行以下命令：

```
syms x                    %符号变量说明
f=taylor(log(1+x))        %f 为函数 log(1+x)的泰勒展式
fccode=ccode(f)           %将 f 转化为 C 语言代码格式
```

则显示结果为：

```
f =
x-1/2*x^2+1/3*x^3-1/4*x^4+1/5*x^5
fccode =
      t0 = x-x*x/2.0+x*x*x/3.0-x*x*x*x/4.0+x*x*x*x*x/5.0;
```

（2）执行以下命令：

```
H=sym(hilb(3))         %H 为 3 阶希尔伯特字符型矩阵
ccodeH=ccode(H)        %获得 H 的 C 语言代码表示形式
```

显示结果为：

```
H =
[   1, 1/2, 1/3]
[ 1/2, 1/3, 1/4]
[ 1/3, 1/4, 1/5]
ccodeH =
   H[0][0] = 1.0;     H[0][1] = 1.0/2.0;     H[0][2] = 1.0/3.0;     H[1][0] = 1.0/2.0;     H[1][1] =
1.0/3.0;   H[1][2] = 1.0/4.0;     H[2][0] = 1.0/3.0;     H[2][1] = 1.0/4.0;     H[2][2] = 1.0/5.0;
```

### 6. fortran 命令

该命令能求得符号表达式的 FORTRAN 语言源代码表达式，其调用格式为：

● fortran(s)：返回符号表达式 $s$ 的 FORTRAN 语言书写格式。

【例 6-11】

（1）执行以下命令：

```
syms x                 %符号变量说明
f=taylor(log(1+x))     %f 为函数 log(1+x)的泰勒展式
ffortran=fortran(f)    %将 f 转化为 FORTRAN 语言代码格式
```

显示结果为：

```
f =
x-1/2*x^2+1/3*x^3-1/4*x^4+1/5*x^5
ffortran =
t0 = x-x**2/2+x**3/3-x**4/4+x**5/5
```

（2）执行以下命令：

```
H=sym(hilb(3))              %H 为 3 阶希尔伯特字符型矩阵
fortranh=fortran(H)         %获得 H 的 FORTRAAN 语言代码表示形式
```

显示结果为：

```
H =
[   1, 1/2, 1/3]
[ 1/2, 1/3, 1/4]
[ 1/3, 1/4, 1/5]
fortranh =
   H(1,1) = 1     H(1,2) = 1.E0/2.E0     H(1,3) = 1.E0/3.E0     H(2,1) = 1.E0/2.E0     H(2,2) =
1.E0/3.E0    H(2,3) = 1.E0/4.E0    H(3,1) = 1.E0/3.E0    H(3,2) = 1.E0/4.E0     H(3,3) = 1.E0/5.E0
```

## 6.2.2 特殊函数

MATLAB 符号工具箱中提供了一些特殊函数的专用命令，本节简要介绍以下特殊函数：

sinint 函数、cosint 函数、zeta 函数以及 lambertw 函数。

### 1．sinint—— 正弦积分函数

正弦积分函数的数学表达式为：$\text{sinint}(x)=\int_0^x \frac{\sin t}{t}dt$。事实上，sinint 命令用 MATLAB 命令很容易就能实现：sinint(x)=int(sin(t)/t,t,0,x)。之所以专门给出 sinint 命令，是为了更为简捷地实现上述功能。

【例 6-12】

执行以下命令：

```
syms w
sinint(w^2/(w-1))
```

则显示结果为：

```
ans =
sinint(w^2/(w-1))
```

### 2．cosint —— 余弦积分函数

余弦积分函数的数学表达式为：$\text{cosint}(x)=\gamma+\log x+\int_0^x \frac{\cos t-1}{t}dt$，其中 $\gamma$=.5772156649015328606065120900824 为欧拉常数。用 MATLAB 命令也很容易实现 cosint 命令所具有的功能：cosint(x)=Gamma+log(x)+int((cos(t)-1)/t,t,0,x)。

### 3．zeta 函数

zeta 函数的数学表达式为：$\text{zeta}(z)=\sum_{k=1}^{+\infty}\frac{1}{k^z}$。用 MATLAB 命令可以描述为 zeta(z)=sum(1/k^z,k,1,inf)。Zeta 函数的调用格式有：

- zeta(z)：在区间[1,+∞]上计算函数 $\frac{1}{k^z}$ 关于变量 k 的积分值。
- zeta(n,z)：计算 zeta(z)的 $n$ 阶导数。

### 4．lambertw 函数

lambertw 函数又称为兰伯特 $W$ 函数。该函数有以下几种调用格式：

w=lambertw(x)：返回值为 $we^w=x$ 的解。

w=lambertw(k,x)：为该多值函数的第 $k$ 部分。

## 6.3　可控精度的实现

MATLAB 共有 3 种针对不同计算精度的算法，分别为：

（1）针对浮点运算的数值算法。

（2）针对精确运算的符号算法。

（3）可控精度的算法。

以上 3 种算法中，第 1 种方法是速度最快的算法，它与 FORTRAN 语言、C 语言中的浮点运算算法完全相同。由于所有的数值在计算机中都是用若干位二进制数来表达的，因此它们在机器内的表达和计算结果都是一个被截断的近似值。

下面来看一个简单的例子：

【例 6-13】

（浮点运算）在 MATLAB 空间中执行命令：

```
a=1/3+1/7
```

显示结果为：

```
a =
    0.4762
```

在计算 $a$ 的过程中，浮点运算总共有 3 处产生了误差：第 1 处发生在计算 $\frac{1}{3}$ 时；第 2 处发生在计算 $\frac{1}{7}$ 时；第 3 处则发生在求和处。

与浮点运算的算法不同，符号算法则一切以计算结果的准确为中心，它完全按照有理数的计算方法进行计算，但它往往比前面讲到的第 1 种、第 3 种算法耗时要多些。

【例 6-14】

（符号运算）用第 2 种算法再来计算 $\frac{1}{3}+\frac{1}{7}$。执行以下命令：

```
a=sym(1/3+1/7)
```

执行结果为：

```
a =
10/21
```

由此可见，在这种情况下，符号运算没有产生任何误差。

第 3 种算法事实上是第 1 种算法和第 2 种算法的一种折中。这种算法可以用命令来控制运算过程中参加运算的各个量取多少位有效数字，从而达到控制计算精度的目的。要注意的是，所取有效数字的多少对运算速度有重要的影响。

可控精度的实现可以借助于 digits 命令和 vpa 命令。

### 1．digits 命令

有以下两种调用格式：

digits：查询 digits 的值。

digits(n)：在该命令之后的计算中取 $n$ 位有效数字。

digits 近似于一个全局变量，其默认值为 32。例如，想查询现在 MATLAB 的有效数字位数是多少，那么输入 DIGITS、Digits 或者 digits 后都会得到相同的显示结果：Digits=32；但是如果在工作空间中已经创建了变量 digits（如 digits=20），则变量 digits 将成为一个普通

变量，这与可控精度的计算没有任何关系了。

【例 6-15】

执行以下命令：

```
Digits          %查询当前空间中有效数字的位数是多少
digits          %功能完全同上。
digits(20)      %将当前空间中有效数字的位数设为 20 位
digits          %再次查询当前空间中有效数字的位数是多少
digits+5        %此时 digits 成为一个普通的变量了
```

显示结果为：

```
Digits = 32
Digits = 32
Digits = 20
ans =
    25
```

### 2. vpa 命令

该命令的功能是在计算精度 Digits 为 $n$ 的情况下进行可控精度计算，其调用格式为：

r=vpa(s)：按 Digits 确定的可控精度计算 $s$ 的值。

vpa(s,d)：指定 $s$ 的精度为 $d$ 位有效数字。

注意：在 vpa 的表达式中，不管参与运算的变量是哪种类型，其计算结果均是符号量。

【例 6-16】

（1）执行命令：

```
vpa(pi,13)
```

显示结果为：

```
ans =
3.141592653590
```

由此可见，此时 π 有 13 位有效数字。

（2）执行命令：

```
clear              %清除工作空间中的所有变量
vpa(hilb(3),5)     %3*3 的希尔伯特矩阵的有效数字位数设为 5 位，返回值为 ans
whos               %详细查询当前工作空间中的变量列表
```

显示结果为：

```
ans =
[      1., .50000, .33333]
[ .50000, .33333, .25000]
[ .33333, .25000, .20000]
```

```
  Name      Size              Bytes   Class
  ans       3x3               960   sym object
Grand total is 59 elements using 960 bytes
```

由此可见，vpa 的返回值是符号型的。

（3）执行以下程序：

```
digits(3);                 %将有效数字位数设为 3 位
a=sym(1/3);                %a 为符号型的"1/3"
b=sym(pi);                 %b 为符号型的"pi"
c=3/8;
result1=vpa(a+b)           %计算 a+b 的值
result2=vpa(a+c)           %计算 a+c 的值
digits(10);                %将有效数字的位数设为 10 位
result3=vpa(a+b)           %计算 a+b 的值
result4=vpa(a+c)           %计算 a+c 的值
```

显示结果为：

```
result1 =
3.47
result2 =
.708
result3 =
3.474925987
result4 =
.7083333333
```

## 6.4 抽象函数的创建

科学计算和工程应用中经常会遇到用 $f(x)$、$g(x,y,z)$或者 $f$、$g$ 等类似的式子来表示一个抽象函数的情况。这些抽象函数在 MATLAB 中可以用两种方法来实现：sym 命令法和 MAPLE 函数库中 map 命令法。

### 1．用 sym 命令创建抽象函数

假设函数$f$是变量 $var1,var2,\cdots,varn$ 的抽象函数(这里的 $vari$ 可以是表达式，其中 $i$=1,2, …, $n$)，则可借助于 sym 命令创建抽象函数$f$。使用格式为：

f=sym('f(var1,var2,...,varn)')

用这种方法创建的抽象函数可以像普通函数一样进行各种运算，如微分、积分。

【例 6-17】

执行以下程序：

```
syms x                          %符号变量说明
f=sym('f(x,y,z)')               %创建抽象函数 f(x,y,z)
g=sym('g(x,x+y,x*y*z,x/y+z)')   %创建抽象函数 g(x,x+y,x*y*z,x/y+z)
```

```
diff_fx=diff(f,x)                 %求抽象函数 f 的微分
diff_gx=diff(g,x)                 %求抽象函数 g 的微分
```

显示结果为：

```
f =
f(x,y,z)
g =
g(x,x+y,x*y*z,x/y+z)
diff_fx =
diff(f(x,y,z),x)
diff_gx =
D[1](g)(x,x+y,x*y*z,x/y+z)+D[2](g)(x,x+y,x*y*z,x/y+z)+D[3](g)(x,x+y,x*y*z,x/y+z)*y*z+D[4](g)(x,x+y,x
*y*z,x/y+z)/y
```

### 2．用 map 命令创建抽象函数

该种方法比较灵活，有两种调用格式：

```
map(fcn,expr,arg1,arg2,…,argn)
map(fcn, arg1,expr, arg2,…,argn)
```

以上命令中，fcn：一个操作手续或者名称；expr：表达式；argi：用于操作的对象。

【例 6-18】

执行以下程序：

```
a=maple('map(f,x,y,z)')           %创建抽象函数 f(x,y,z)
b=maple('map(f,y+z,z)')           %创建抽象函数 f(y,z)+f(z,z)
c=maple('map(g,x,y/z,z)')         %创建抽象函数 g(x,y/z,z)
d=maple('map(f,{x,y,z})')         %创建抽象函数{f(x),f(y),f(z)}
e=maple('map(x->x^2,x+y)')        %创建抽象函数 x^2+y^2
```

显示结果为：

```
a =
f(x,y,z)
b =
f(y,z)+f(z,z)
c =
g(x,y/z,z)
d =
{f(x), f(y), f(z)}
e =
x^2+y^2
```

## 6.5 如何在 MATLAB 中使用 MAPLE

MATLAB 的符号运算功能全都来自于 MAPLE V，本书第 2~5 章所介绍的命令只不过是

其中的一部分而已。如果想深入学习 MATLAB，则需要掌握 MPALE 命令的具体使用方法。

### 6.5.1 访问 MAPLE

MAPLE 命令是非常多的，使用起来也十分灵活。限于篇幅，这里只介绍两种最常用的使用格式：maple(statement)和 maple('fuction',arg1,arg2,…)。

#### 1. maple(statement)

该命令的功能是把对变量、表达式以及函数等的描述（statement）传递到 MATLAB 的符号运算引擎——MAPLE V，由 MAPLE 进行计算并返回字符型结果。值得一提的是，这个命令的功能十分强大，它可调用 MAPLE 函数库中除了图像处理函数外的所有函数。

【例 6-19】

s=evalf(num,n)是 MAPLE 中类似 MATLAB 中的 vpa 命令的一个命令，它的作用是把数值 *num* 用 *n* 位数字表示出来。

执行以下命令：

```
clear;                        %清除工作空间变量
a=maple('evalf(pi,9)')        %把 pi 用 9 位数字表示出来
b=maple('evalf(1/3,6)')       %把 1/3 用 6 位数字表示出来
c=maple('evalf(1.2,7)')       %把 1.2 用 7 位数字表示出来
whos;                         %详细查看工作空间中的变量
```

执行以上命令的显示结果为：

```
a =
3.14159265
b =
.333333
c =
1.2

  Name      Size            Bytes  Class
  a         1x10               20  char array
  b         1x7                14  char array
  c         1x3                 6  char array
Grand total is 20 elements using 40 bytes
```

可见所得到的 *a,b,c* 均是字符型数组。其中变量 *c* 只有 3 位数字，这是因为 1.2 本身就是一个精确表达的数。

#### 2. maple('fuction',arg1,arg2,…)

该命令是 MATLAB 调用 MAPLE 函数库中的函数 function 所用的标准命令。其中：function 是函数的名称；arg1,arg2,…是函数 function 所用的函数。

【例 6-20】

执行以下命令：

```
f=4/3;
g='1/3';
h=sym('1.234*4');
mf=maple('evalf',f,7)          %把 f 用 7 位数表示出来
mg=maple('evalf',g,5)          %把 g 用 5 为数表示出来
mh=maple('evalf',h,3)          %把 h 用 3 位数表示出来
```

显示结果为：

```
mf =
1.333333
mg =
.33333
 mh =
 4.94
```

执行以下命令：

```
syms x                         %符号说明
maple('bernoulli',5,x)         %x 的 5 阶贝努立表达式
```

显示结果为：

```
ans =
-1/6*x+5/3*x^3+x^5-5/2*x^4
```

### 6.5.2 MAPLE V 中的特殊函数及其调用方法

MATLAB 提供了 50 多个特殊函数，但在 MAPLE V 的函数库中还有 40 多个不同的函数可供调用。这些函数进一步体现了 MATLAB 在计算特殊函数方面的强大计算功能。借助于 MAPLE V 的这一强力“引擎”，MATLAB 既可以完成复平面上的误差分析又可以完成 Fresnel 余弦积分一类的复杂运算。

MAPLE V 提供的特殊函数如表 6-1 所示，以供用户使用。

表 6-1 MAPLE V 提供的特殊函数

| 函 数 名 | 所 需 参 数 | 说 明 |
| --- | --- | --- |
| bernoulli | N | 贝努立数 |
| bernoulli | n,z | 贝努立多项式 |
| besseli | x1,x | 第一类贝塞尔函数 |
| besselj | x1,x | 第一类贝塞尔函数 |
| besselk | x1,x | 第二类贝塞尔函数 |
| bessely | x1,x | 第二类贝塞尔函数 |
| beta | z1,z2 | β函数 |
| binomial | x1,x2 | 二项式系数 |
| legendreKc | x | 第一类完全椭圆积分 |
| legendreEc | x | 第二类完全椭圆积分 |
| legendrePic | x1,x | 第三类完全椭圆积分 |

（续）

| 函 数 名 | 所 需 参 数 | 说 明 |
|---|---|---|
| legendreKc1 | x | 使用补系数法计算的第一类完全椭圆积分 |
| legendreEc1 | x | 使用补系数法计算的第二类完全椭圆积分 |
| legendrePic1 | x1,x | 使用补系数法计算的第三类完全椭圆积分 |
| erfc | z | 补误差函数 |
| erfc | z,n | 迭带的补误差函数 |
| Ci | z | 余弦积分 |
| Si | z | 正弦积分 |
| Chi | z | 双曲余弦积分 |
| Shi | z | 双曲正弦积分 |
| Li | x | 对数积分 |
| Ssi | z | 切换的正弦积分 |
| dawson | x | dawson 积分 |
| Psi | z | Digamma 函数 |
| dilog | x | 双对数积分 |
| erf | z | 误差函数 |
| euler | n | 欧拉数 |
| euler | n,z | 欧拉多项式 |
| ei | n | 指数积分 |
| ei | n,z | 指数积分 |
| fresnelC | x | 菲涅尔（fresnel）余弦积分 |
| fresnelS | x | 菲涅尔（fresnel）正弦积分 |
| gamma | z | γ函数 |
| gamma | z1，z2 | 非完全γ函数 |
| lngamma | z | 对数γ函数 |
| harmonic | n | 调和函数 |
| hypergeom | x1,x2 | （广义的）超几何函数 |
| legendreF | x1,x | 第一类非完全椭圆积分 |
| legendreE | x1,x | 第二类非完全椭圆积分 |
| legendrePi | x1,x2,x | 第三类非完全椭圆积分 |
| zeta | z | 黎曼ζ（zeta）函数 |
| zeta | n,z | 黎曼ζ（zeta）函数 |
| zeta | n,z,x | 黎曼ζ（zeta）函数 |
| T | n,x | 第一类切比雪夫多项式 |
| U | n,x | 第二类切比雪夫多项式 |
| G | n,x1,x | Gegenbauer 多项式 |
| H | n,x | 埃米尔特多项式 |
| P | n,x1,x2,x | 雅可比多项式 |
| L | n,x | 拉盖尔多项式 |
| L | n,x1,x | 广义拉盖尔多项式 |
| P | n,x | 勒让德多项式 |

上面的特殊函数中，MAPLE 命令只能调用其中的几个。利用 mfun 函数则可全部调用上述命令。mfun 函数可以计算 MAPLE 函数库中特殊函数的数值结果，其调用格式为：

mfun('function',p1,p2,…,pk)：其中 function 为 MAPLE 函数库中的特殊函数名；p1,p2,…,pk 为计算特殊函数 function 时所需的参数。这些参数中除了最后一个参数 pk 可以是矩阵以外，其他所有参数都必须严格遵守 MAPLE V 中调用该函数时相应的数据类型的规定。

【例 6-21】

执行以下命令：

```
yb=mfun('bernoulli',10,[3 5])    %计算 5 阶贝努立式 x=3,x=5 处的值
yf=mfun('FresnelC',2.5)          %计算菲涅尔余弦积分
ych=mfun('Chi',4)                %计算双曲余弦积分
ye=mfun('Ei',[1.5,2;2.5,3])      %计算指数积分在矩阵[1.5,2;2.5,3]处的值
执行以上命令的显示结果为：
yb =
  1.0e+006 *
    0.0051    2.8234
yf =
    0.4574
ych =
    9.8135
ye =
    3.3013    4.9542
    7.0738    9.9338
```

## 6.6 函数计算器及泰勒计算器的使用

MATLAB 6.x 和 MATLAB 5.3 一样，为用户提供了两个图形界面化的函数操作计算器：函数计算器 funtool 和泰勒计算器 taylortool。利用前者可以方便地研究两个一元函数的各种操作及相应的图形变化情况；而利用后者则可以方便地研究一元函数的泰勒展式。

### 6.6.1 函数计算器的使用

在 MATLAB 工作空间中键入命令 funtool，则打开一个对一元函数进行操作的函数计算器。计算器上共有两个图形窗口（Figure No.1 及 Figure No.2）和一个控制窗口，如图 6-2、6-3 和 6-4 所示。其中图 6-2 和图 6-3 所示的窗口分别用来演示函数 $f$ 和函数 $g$ 的图形。在任何情况下三个窗口中只有一个窗口处于激活状态（只有当刚进行函数计算器界面时，三个函数窗口才都处于激活状态）。

控制窗口中，控制栏“f=”、“g=”分别用来书写函数 $f(x)$,$g(x)$的表达式。控制栏“x=”用于设定函数自变量的变化范围；标有“a=”的控制栏可用来指定参与运算的参数 $a$ 的值。每次打开函数计算器时，默认 f=x、g=1、x=[-2*pi,2*pi]、a=1/2。

【例 6-22】

打开函数计算器（在 MATLAB 工作空间中键入 funtool 然后按 Enter 键即可），在控制窗口的“f=”控制栏中键入 sin(x)按 Enter 键，则图 6-2 所示窗口内容马上变为如图 6-5 所示。

在“g=”控制栏中键入 cos(x)按 Enter 键，则图 6-3 所示窗口内容马上变为如图 6-6 所示。把“x=控制栏”改为[0,2*pi]按 Enter 键，图 6-5 和图 6-6 窗口中内容分别变为如图 6-7、图 6-8 所示。

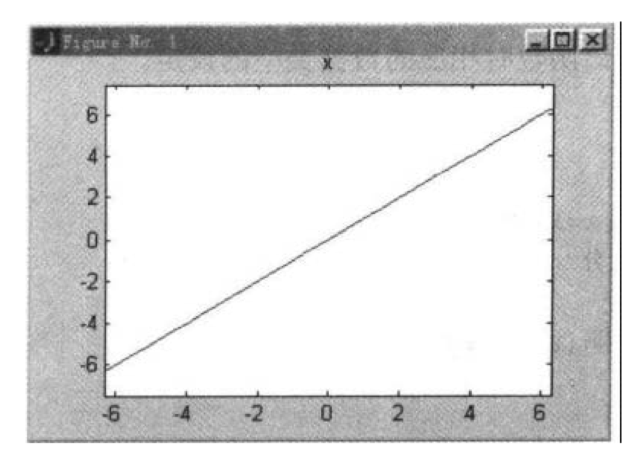



图 6-2 函数计算器图形窗口 1

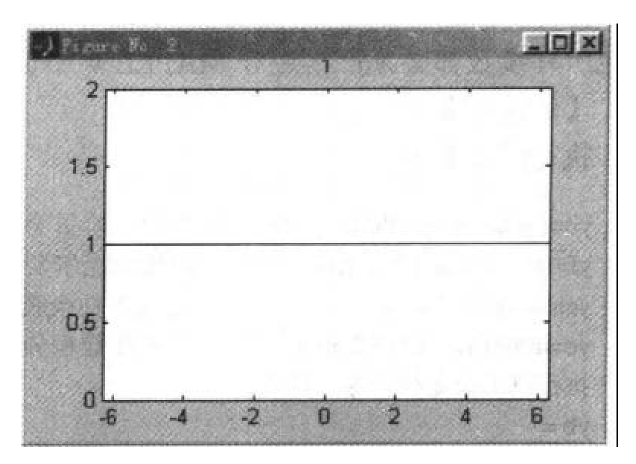



图 6-3 函数计算器图形窗口 2

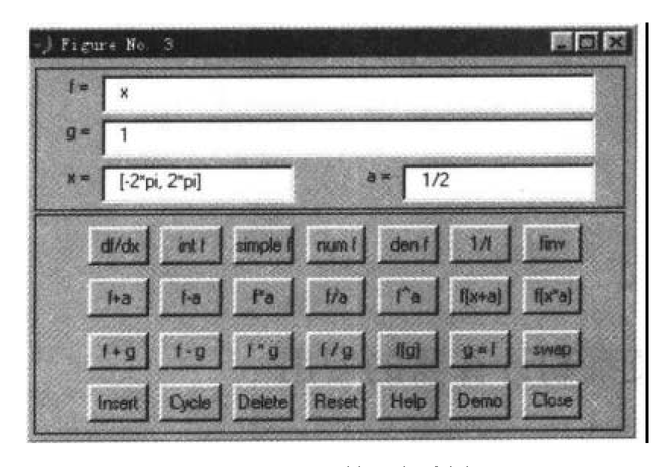



图 6-4 函数计算器控制窗口

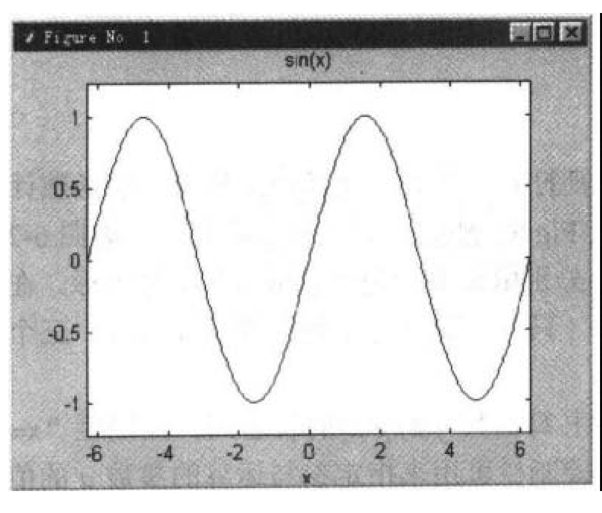



图 6-5 sin(x)的图形

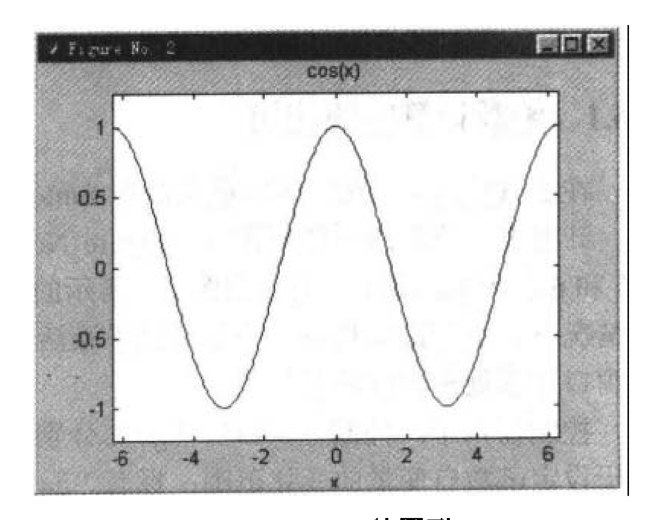



图 6-6 cos(x)的图形

图 6-4 所示控制窗口的下半部分是运算及帮助演示部分，共有 4 排按钮。其中第 1 排 7 个按钮的作用分别是：

dt/dx 求函数 $f(x)$的导数。
int f 求函数 $f(x)$的积分。
simple f 对函数 $f(x)$化简。
num f 求函数 $f(x)$的分子部分。
den f 求函数 $f(x)$的分母部分。
1/f 求函数 $f(x)$的倒数。
finv 求函数 $f(x)$的反函数。

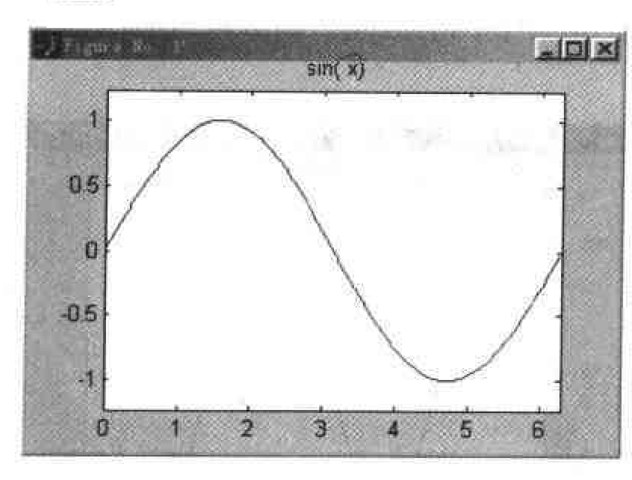



图 6-7　区间[0,2*pi]上 sin(x)的图形

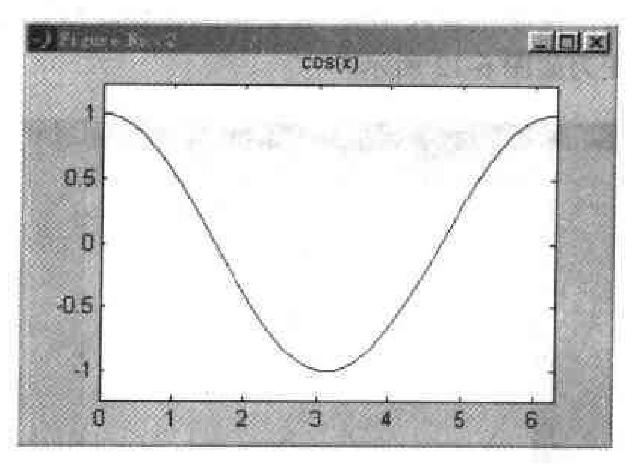



图 6-8　区间[0,2*pi]上 cos(x)的图形

【例 6-23】

若在函数计算器中的“f=”控制栏输入 sin(x) 后按 Enter 键，其余控制栏均取默认项，用鼠标单击▓键则图 6-2 所示的窗口内容变为如图 6-9 所示。在“f=”控制栏输入 sin(x) 后按 Enter 键，其余控制栏均取默认项，用鼠标单击▓键则图 6-2 所示窗口内容变为如图 6-10 所示。

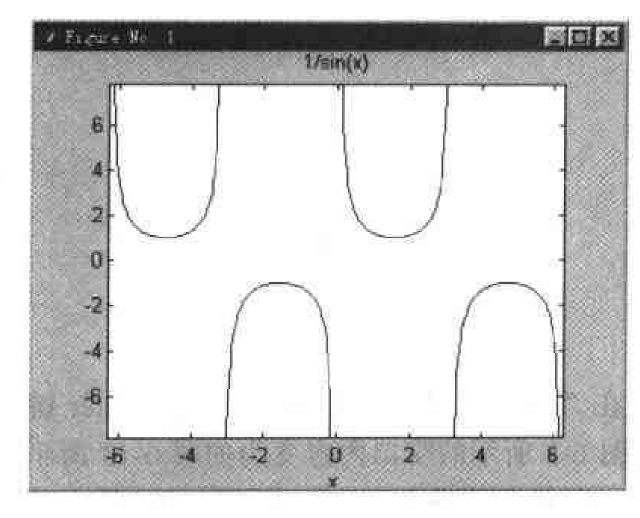



图 6-9　由 1/f 键得到的函数图形

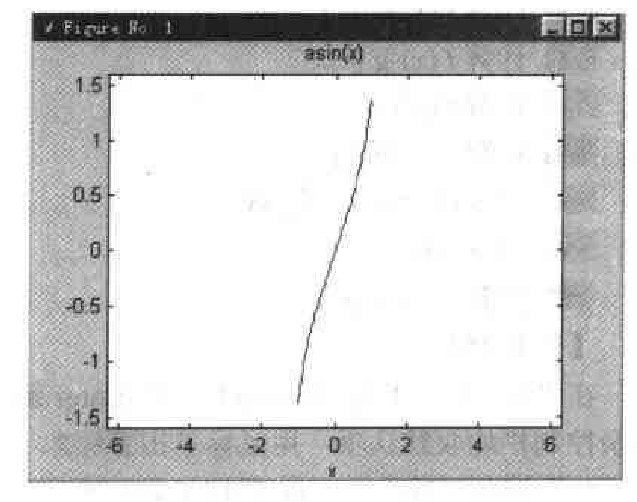



图 6-10　由 finv 键得到的函数图形

图 6-4 所示控制窗口的下半部分，第 2 排 7 个按钮的作用分别如下：

f+a 计算 $f(x)+a$。
f-a 计算 $f(x)-a$。
f*a 计算 $af(x)$。

计算 $f(x)/a$。

计算 $f^{a}(x)$。

计算 $f(x+a)$。

计算 $f(x*a)$。

【例 6-24】

在“f=”控制栏输入 sin(x) 后按 Enter 键，其余控制栏均取默认项，用鼠标单击键，则图 6-1 所示的窗口内容变为如图 6-11 所示。在“f=”控制栏输入 sin(x) 后按 Enter 键，在“a=”控制栏输入 1/3，其余控制栏均取默认项，用鼠标单击键，则图 6-2 所示的窗口内容变为如图 6-12 所示。

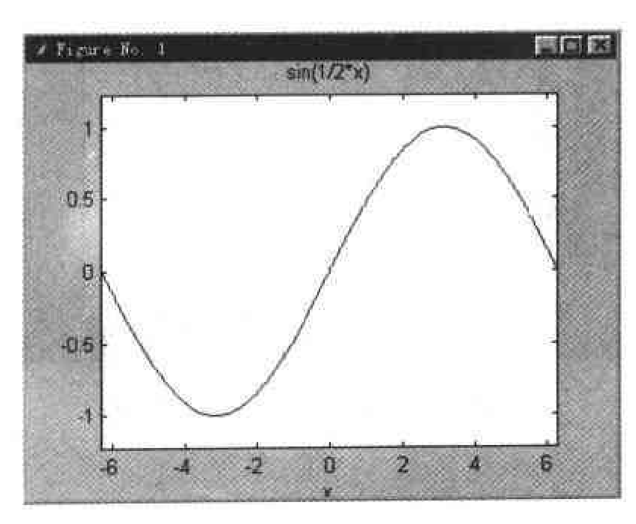



图 6-11 由 f(x*a)键得到的函数图形

图 6-12 由 f(x+a)键得到的函数图形

图 6-4 所示控制窗口的下半部分，第 3 排 7 个按钮的作用分别如下：

计算 $f(x)+g(x)$。

计算 $f(x)-g(x)$。

计算 $f(x)g(x)$。

计算 $f(x)/g(x)$。

计算符合函数 $f(g(x))$。

令 $g(x)=f(x)$。

交换 $f(x)$和 $g(x)$。

【例 6-25】

在“f=”控制栏输入 sin(x) 后按 Enter 键，在“g=”控制栏输入 cos(x) 后按 Enter 键，其余控制栏均取默认项，用鼠标单击键，则图 6-2 所示的窗口内容变为如图 6-13 所示。在“f=”控制栏输入 sin(x) 后按 Enter 键，在“g=”控制栏输入 cos(x) 后按 Enter 键，其余控制栏均取默认项，用鼠标单击键，则图 6-2 所示的窗口内容变为如图 6-14 所示。

【例 6-26】

在“f=”控制栏输入 sin(x)后按 Enter 键，在“g=”控制栏输入 cos(x) 后按 Enter 键，其余控制栏均取默认项，则图 6-2、图 6-3 所示的窗口内容变为如图 6-15、图 6-16 所示。用鼠标单击键，则图 6-15、图 6-16 所示的窗口内容变为如图 6-17、图 6-18 所示。

最后一行 7 个的作用是：

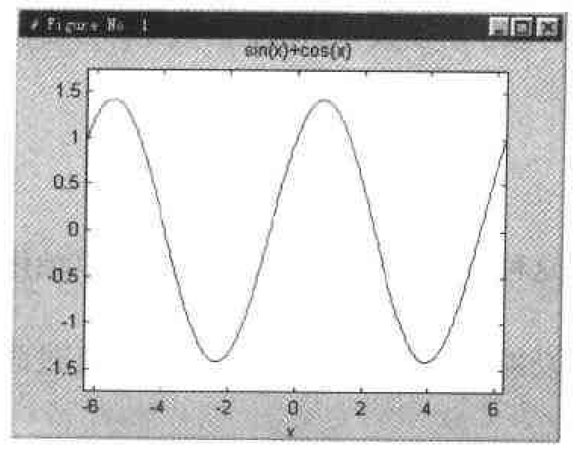



图 6-13　由 f+g 键得到的函数图形

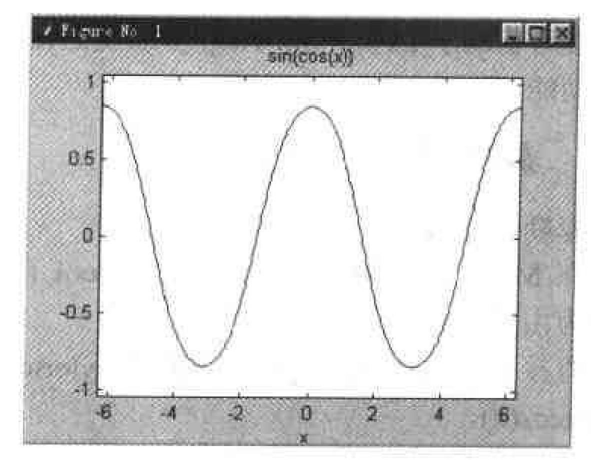



图 6-14　由 f(g)键得到的函数图形

Insert 把当前函数 $f(x)$插入到计算器所包含的典型函数演示表中。

Cycle 在图 6-1 中顺序显示典型函数演示表中的函数曲线。

Delete 把当前图 6-1 中的函数 $f(x)$从典型函数演示表中删掉。

Reset 将函数计算器置到初始状态。

Help 函数计算器的帮助文件。

Demo 系统自动演示函数计算器的计算功能。

Close 关闭函数计算器。

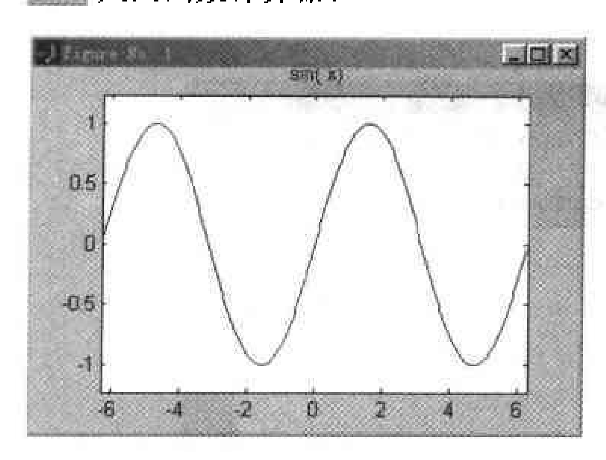



图 6-15　由 f=sin(x)得到的函数图形

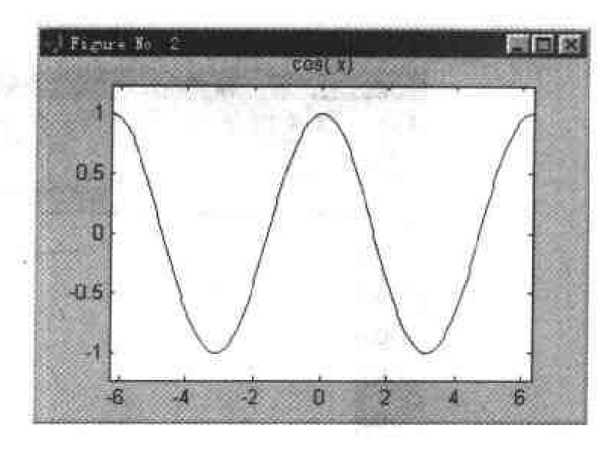



图 6-16　由 g(x)=cos(x)得到的函数图形

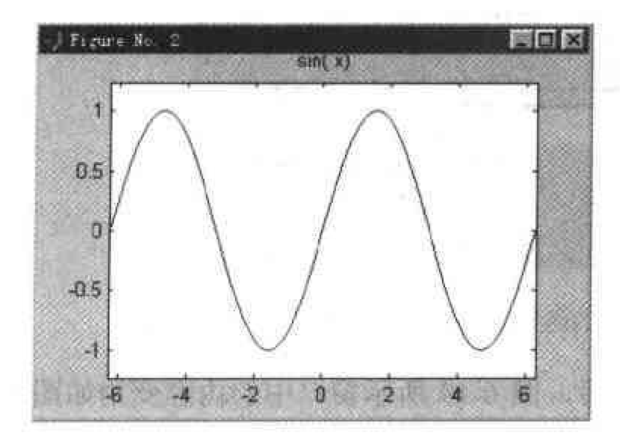



图 6-17　由图 6-15 经过 swap 命令得到的函数图形

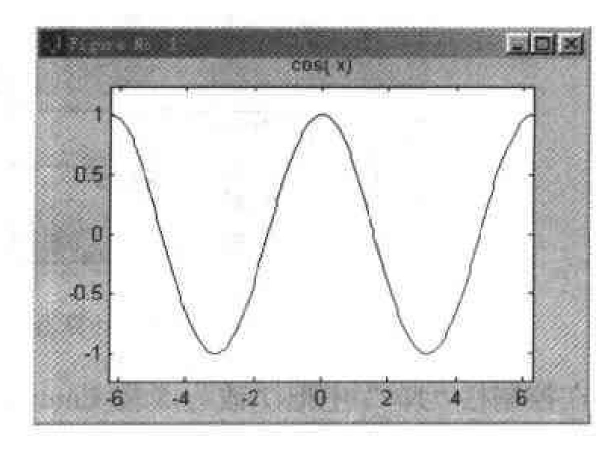



图 6-18　由图 6-16 经过 swap 命令得到的函数图形

读者借助于 Demo 键、Help 键以及 Cycle 键可以了解函数计算器在函数计算方面较为强大的功能。这里不再一一列举。

### 6.6.2 泰勒计算器的使用

泰勒计算器的调用格式有两种：

在 MATLAB 命令窗口键入 taylortool ('f')，这种调用格式会把泰勒计算器的初始化函数指定为由 $f$ 表达的一元函数。

直接在 MATLAB 命令窗口键入 taylortool，功能和上面的调用格式相似，只不过此时默认 f=xcos(x)。

【例 6-27】

（1）直接在 MATLAB 命令窗口键入 taylortool('x^3*sin(x)')，按 Enter 键后会得到图 6-19 所示的窗口。其中控制栏"f(x)="表示在 MATLAB 命令窗口键入的 taylortool('x^3*sin(x)') 中的函数表达式；控制栏"N="表示泰勒展式的展开阶数，默认值为 7；控制栏"a="表示将函数 $f(x)$在 $x=a$ 处展开，其默认值为 0；控制栏"<x<"表示自变量的取值范围，默认值为[-2π,2π]。图 6-19 中的实线（兰色线条）表示函数 $f(x)$的图形，而虚线（红色线条）表示相应的泰勒展式的图形。从图中可以看出，当自变量 $x$ 较小时，只要泰勒展式的次数较高，那么函数的图形和相应的泰勒展式还是吻合得比较好的。当然这个结论也随着函数表达式和展开次数的不同而不同。另外，从图 6-19 中还可以直接看到泰勒展式的代数表达式为 $T_N(x) = x^4 - 1/6\,x^6$。

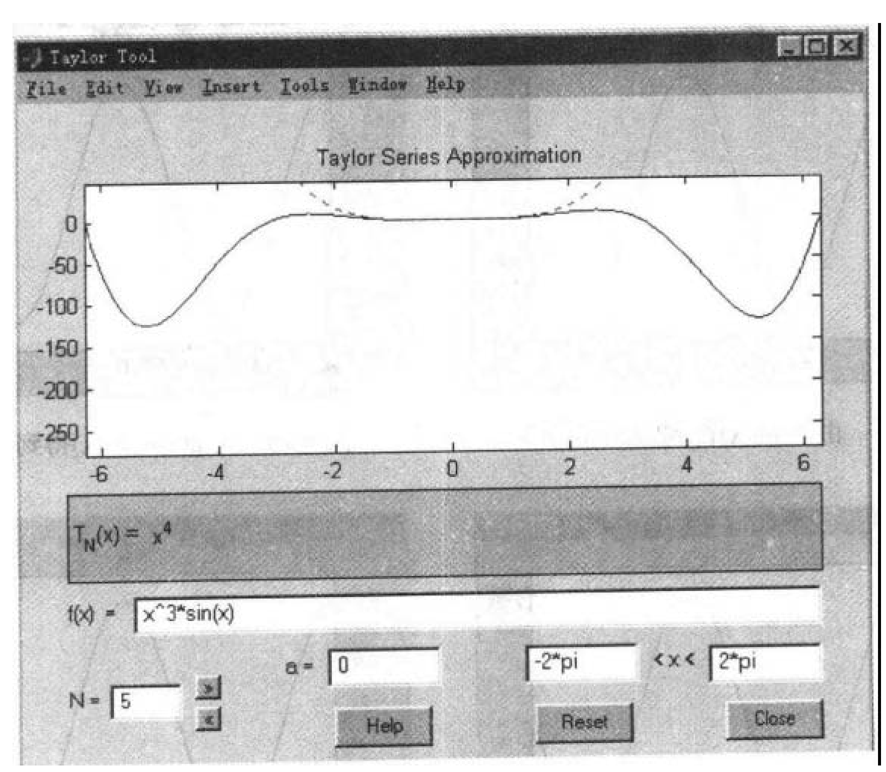



图 6-19 由泰勒计算器得到的函数 $f(x)=x^3\sin(x)$图形

在控制栏"N="中把 7 改为 5 按 Enter 键后，图 6-19 所示窗口中的内容变为如图 6-20 所示。

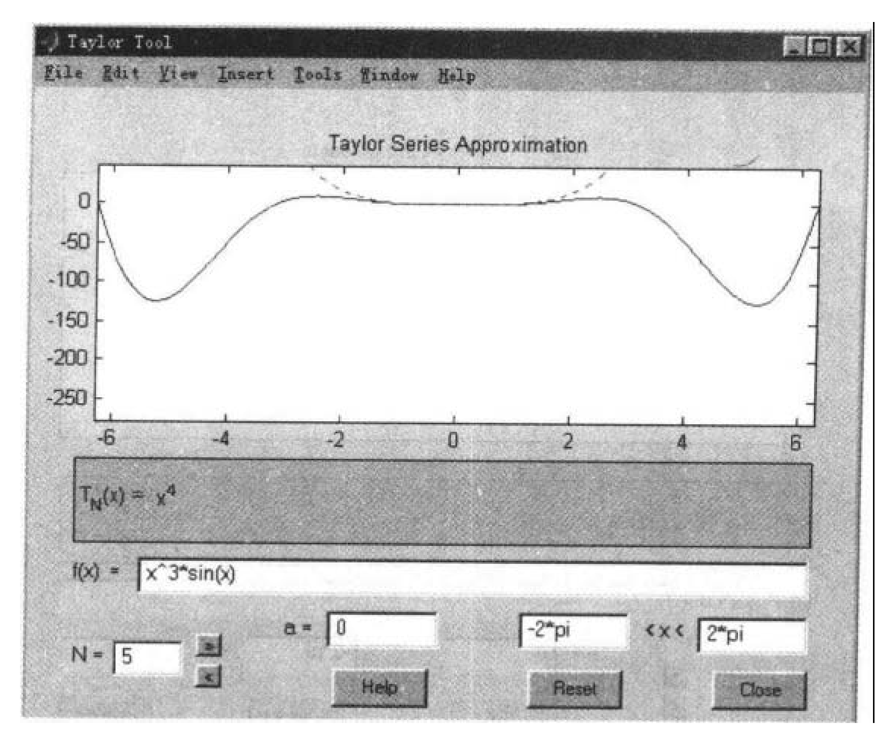



图 6-20 函数 $f(x)=x^3\sin(x)$的 5 阶泰勒展式图形

在控制栏“N=”键入 8，在“a=”控制栏键入 2，在“<x<”控制栏键入-10,10。则图 6-19 所示窗口中的内容变为如图 6-21 所示。

（2）直接在 MATLAB 命令窗口键入 taylortool，按 Enter 键后会得到图 6-22 所示的窗口内容。在“a=”控制栏中输入 pi 按 Enter 键后会得到如图 6-23 所示的图形。至于其他窗口的操作与上面（1）中的操作完全相同，这里不再赘述。

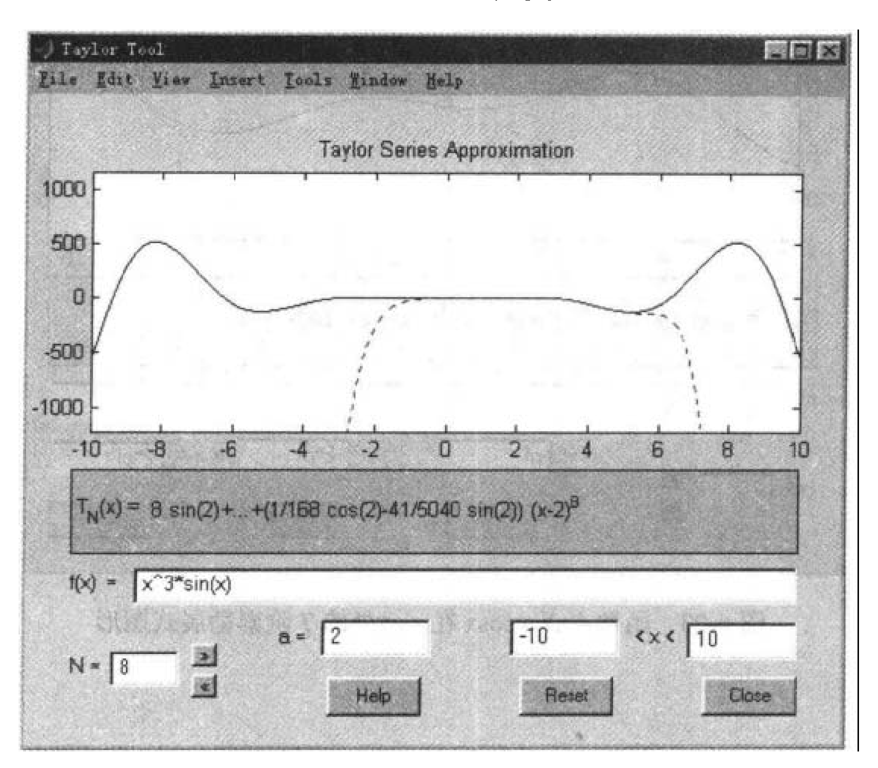



图 6-21 区间[-10,10]上函数 $f(x)= x^3\sin(x)$在 x=2 处的 8 阶泰勒展式图形

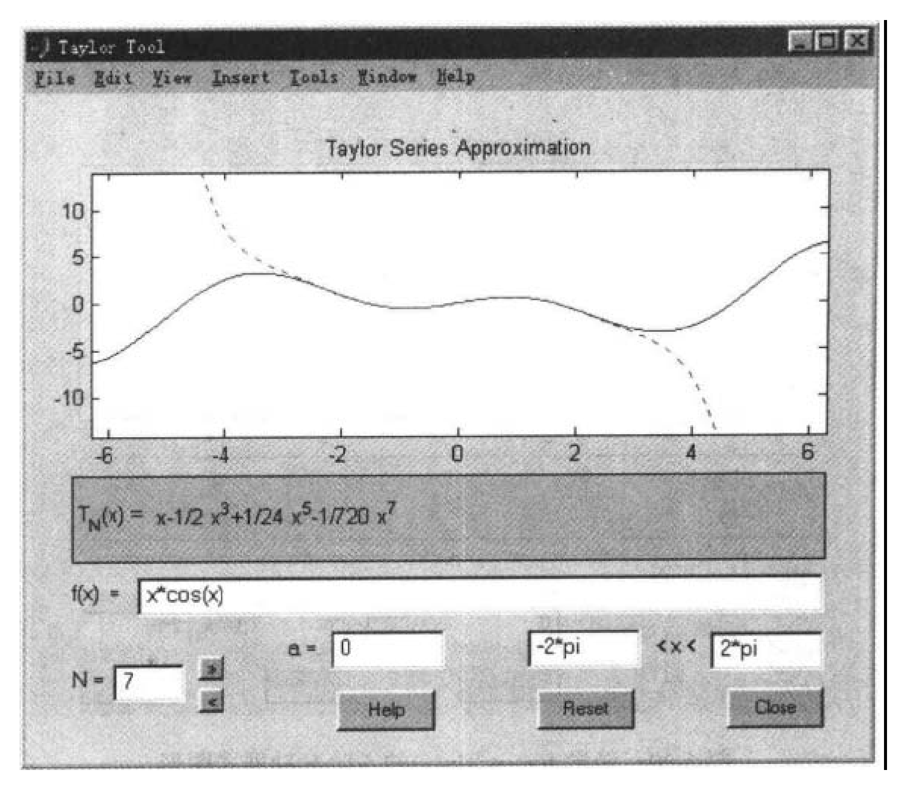



图 6-22 泰勒计算器的默认图形界面

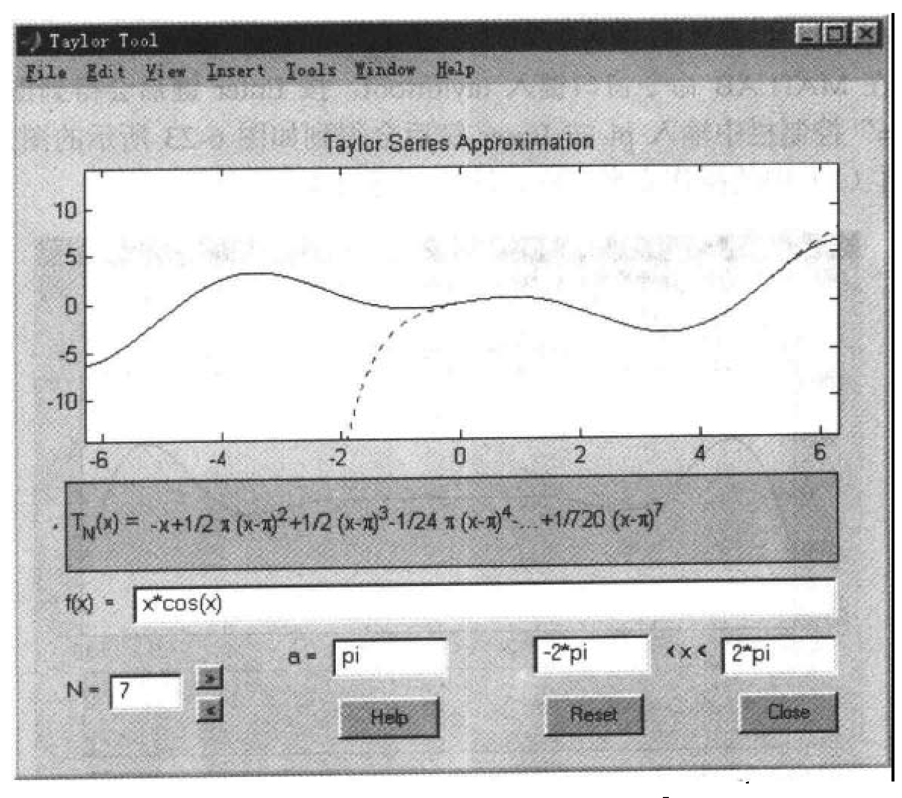



图 6-23 函数 f(x)=xcosx 在 x=π处的 7 阶泰勒展式图形

## 6.7 小结

本章主要介绍了如何用 MATLAB 实现积分变换等内容。可以说，用 MATLAB 几乎可

以解决常见的一切数学问题，包括求解抽象函数形式的数学表达式。本章简要介绍了在MATLAB中如何使用MAPLE的具体方法，这一内容是MATLAB符号运算功能的一个延伸。MATLAB之所以有如此强大的符号运算功能，完全要归功于MAPLE。Maths公司拥有MAPLE的内核后，MATLAB的功能得到了巨大的增强，起到了如虎添翼的作用，这是Maths公司成功运作的一记经典手笔。另外，MATLAB提供的函数计算器以及泰勒计算器使用起来异常方便，读者应当灵活掌握。希望读者能熟练掌握本章所讲述的所有内容。

# 第 7 章　符号函数图形的绘制

知识点：

- fplot 命令
- ezplot 命令
- ezpolar 命令
- ezplot3 命令
- ezmesh 命令
- ezcontour 命令
- ezsurf 命令
- 图形的控制与标注

本章介绍如何在 MATLAB 中绘制符号函数的图形。其中不仅重点介绍 fplot 命令、ezplot 命令、ezpolar 命令、ezplot3 命令、ezmesh 命令、ezcontour 命令、ezsurf 命令、ezmeshc 命令、ezcontourf 命令以及 ezsurfc 命令的使用方法，还介绍如何对图形进行控制，如何标注图形等相关知识。通过对本章的学习，读者会发现 MATLAB 在图形处理方面具有自己的独特性和实用性。

## 7.1　基本绘图命令

MATLAB 提供的 plot 命令以及 plot3 命令是最为常见的二维图形和三维图形绘制命令。在使用 plot 命令以及 plot3 命令绘图时，MATLAB 是把外部输入的数值矩阵转化为图形。由于往往事先无法知道函数应有的图形样式，因此，一般都是采用等步长绘制图形。实际上，很多函数的图形局部变化可能很剧烈，这时若使用 plot 命令或 plot3 命令绘制图形就会失真。因此，绘制图形时，在图形变化剧烈处最好采用小步长，而在图形变化平缓之处最好使用大步长，加快绘图速度，节约内存的占用情况。为此，MATLAB 提供了符号函数绘图命令：fplot 和 ezplot 以及其他绘图命令。它们都能很方便地绘制一元符号函数的图形。

### 7.1.1　fplot 命令

fplot 命令的调用格式主要有：

- fplot(fun,lims,str,tol)：直接绘制函数 $y$=fun($x$)的图形。其中，lims 为一个向量，若 lims 只包含两个元素则表示 $x$ 轴的范围：[xmin,xmax]。若 lims 包含四个元素则前两个元素表示 $x$ 轴的范围：[xmin,xmax]，后两个元素代表 $y$ 轴的范围：[ymin,ymax]；str 可以指定图形的线型和颜色。tol 的值小于 1，代表相对误差，默认值为 0.002，即 0.2%。该命令中 str 和 tol 都是可选项。

- fplot(fun,lims,n)：用最少为 $n+1$ 个点来绘制函数 fun 的图形，其中 $n\geq 1$。默认情况下，$n=1$。最大的步长为(1/n)*(xmax-xmin)。

[X,Y]=fplot(fun,lims,...)：返回 $X$ 和 $Y$，并且 $X$ 和 $Y$ 满足 $Y$=fun($X$)。该命令并不绘制图形。

- [...]=fplot(fun,lims,tol,n,'LineSpec',p1,p2,...)：允许将参数 $p1$、$p2$、...直接传递给函数 fun，即 $Y$=fun($X$,$p1$,$p2$,...)。

☞ 提示：fplot 命令中的函数 fun 既可以是符号表达式，也可以是自己编写的 m 函数。

【例 7-1】

MATLAB 提供了一个名为 humps 的内置函数，其数学表达式为：

humps($x$)=$\dfrac{1}{(x-0.3)^2+0.01}+\dfrac{1}{(x-0.9)^2+0.04}-6$。试分别用 plot 命令和 fplot 命令绘制 humps($x$)在区间[-1,5]上的图形。

方法 1：使用 plot 命令

```
x=-1:0.1:5;
plot(x,humps(x))
```

使用上述命令绘制出的图形如图 7-1 所示。

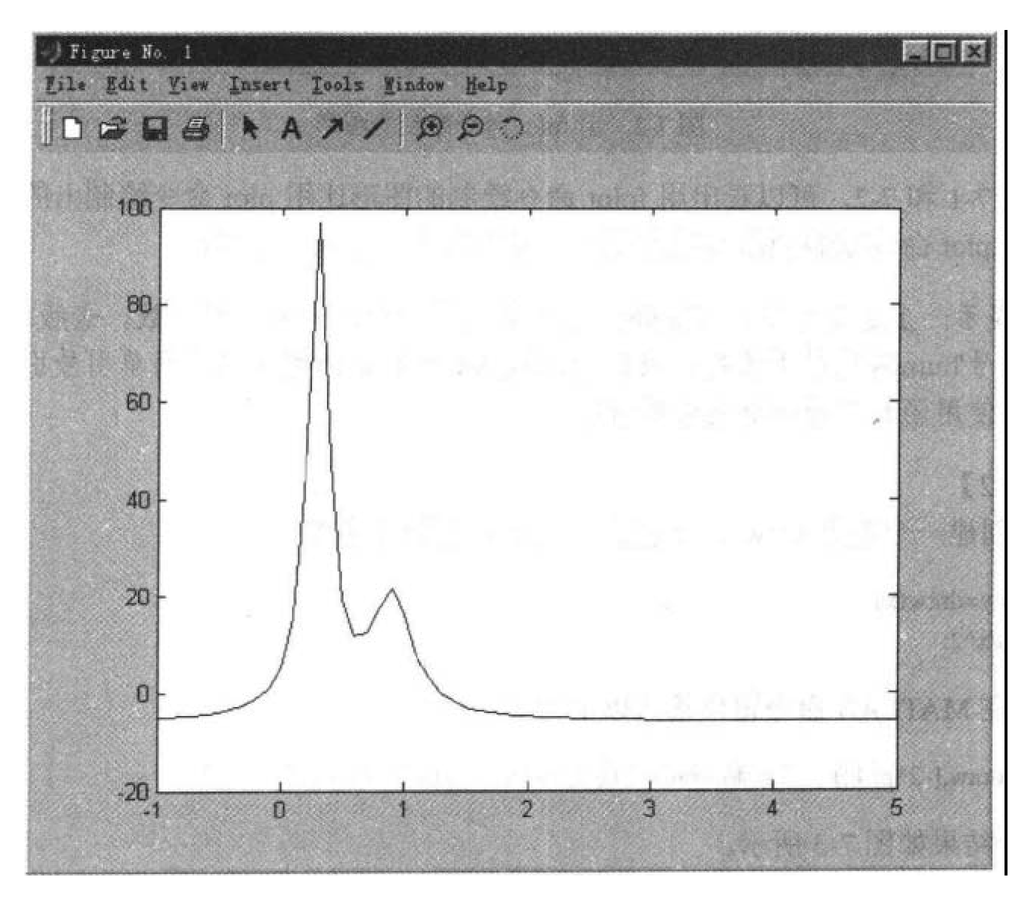



图 7-1　用 plot 命令绘制的图形

方法 2：使用 fplot 命令

```
fplot(@humps,[-1 5])
```

使用上述命令所绘制的图形如图 7-2 所示。

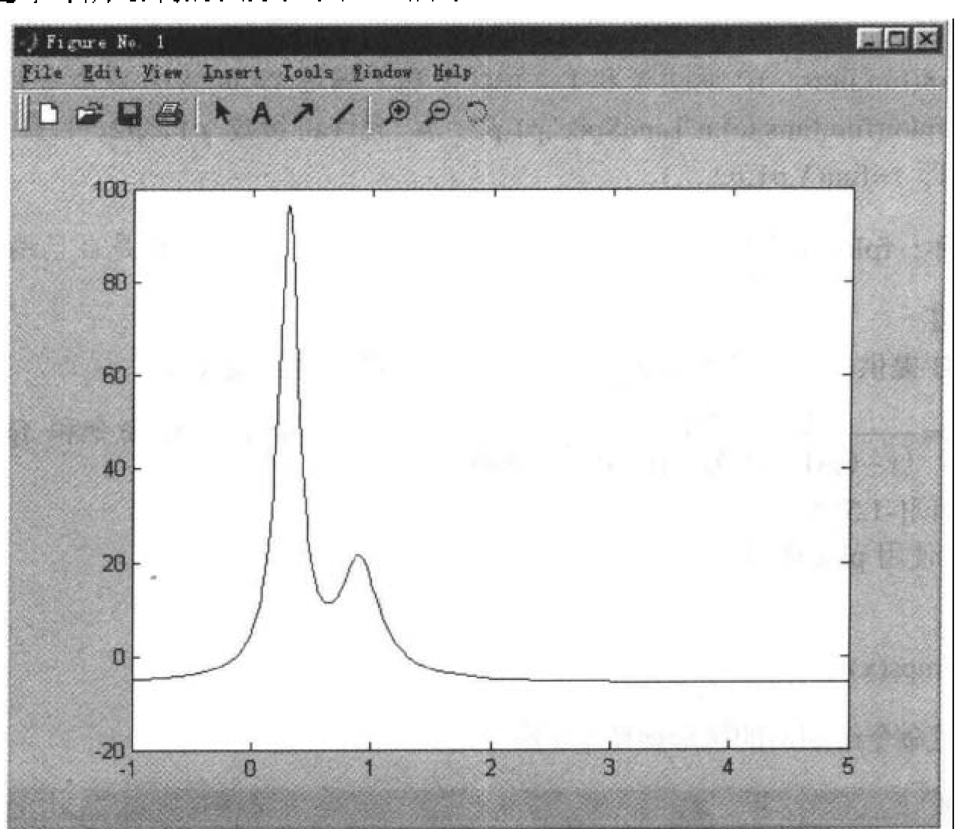



图 7-2 用 fplot 命令绘制的图形

对比图 7-1 和 7-2，可以看出用 fplot 命令绘制的图形比用 plot 命令绘制出的图形要光滑些，要想使 plot 命令绘制的图形更为光滑，可以缩小自变量 $x$ 的步长。

☞ 提示：上述命令中，@humps 表示以函数句柄的形式引用函数，老版本 MATLAB 中使用单引号'humps'来引用函数，虽然 MATLAB 6.x 中仍然可以使用单引号引用函数，但发展趋势是使用函数句柄的形式引用函数。

【例 7-2】

（1）创建一个名为 draw 的 m 函数。可以执行以下命令：

```
function y=draw(x)
y=sin(x)/x^2;
```

（2）在 MATLAB 命令窗口输入以下命令：

```
fplot(@draw,[-2*pi 1])   %绘制 draw 函数在区间[-2π,1]上的图形
```

则执行结果如图 7-3 所示。

（3）直接在 MATLAB 命令窗口输入以下命令：

```
fplot('sin(x)/x^2',[-2*pi 1])
%直接在图形窗口绘制函数 sin(x)/x^2 在区间[-2*pi,1]上的图形
```

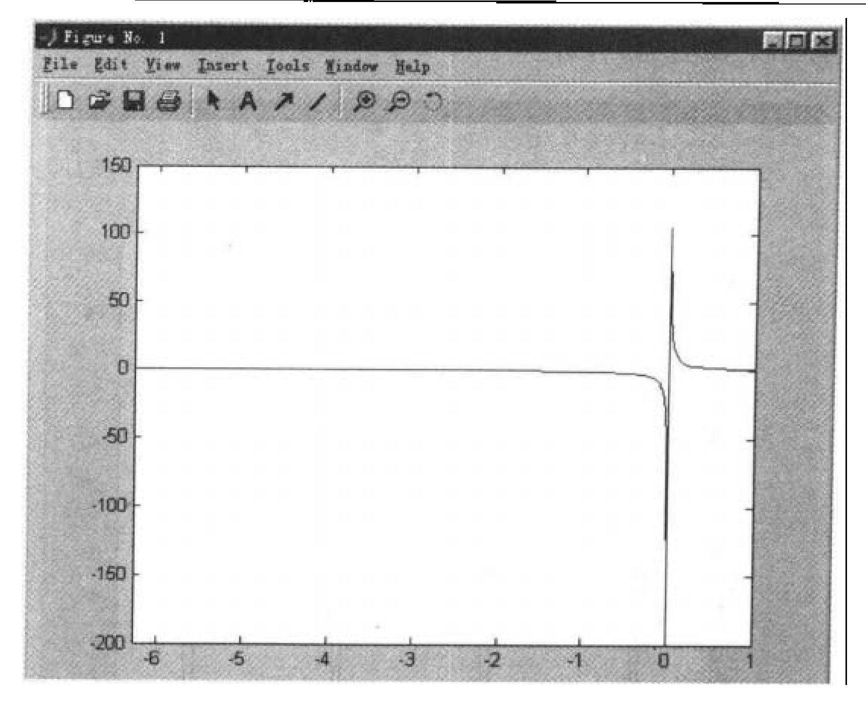



图 7-3　利用 m 函数绘制的图形

执行以上命令得到的结果如图 7-4 所示。

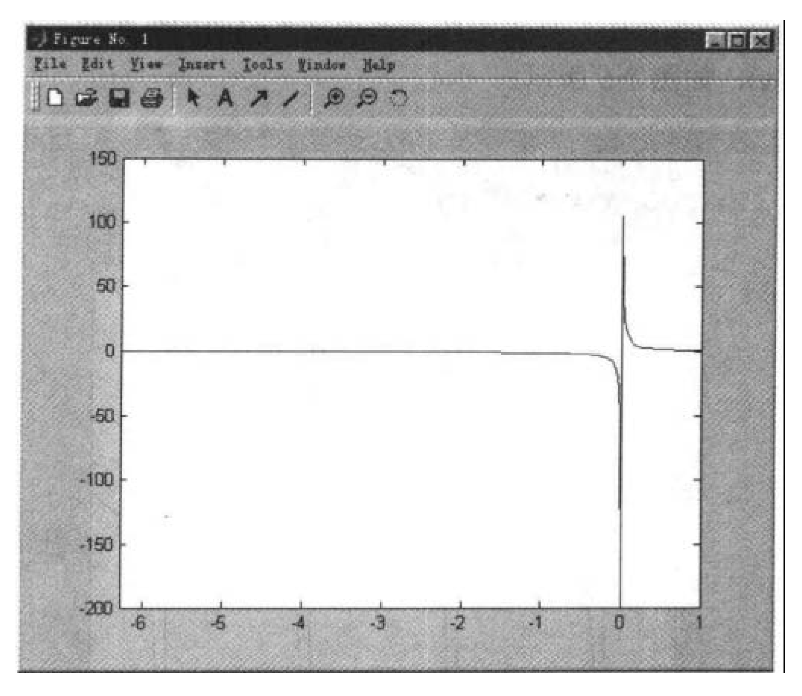



图 7-4　直接由函数表达式绘制的图形

由此可见采用上述两种方法绘制的图形是一模一样的。

【例 7-3】

试在区间 $x\in[-2\pi,2\pi]$，$y\in[-2\pi,2\pi]$上绘制 tan($x$)、sin($x$)和 cos($x$)的图形。

执行命令：

```
fplot('[tan(x),sin(x),cos(x)]',2*pi*[-1 1 -1 1])
```

以上程序的执行结果如图 7-5 所示。

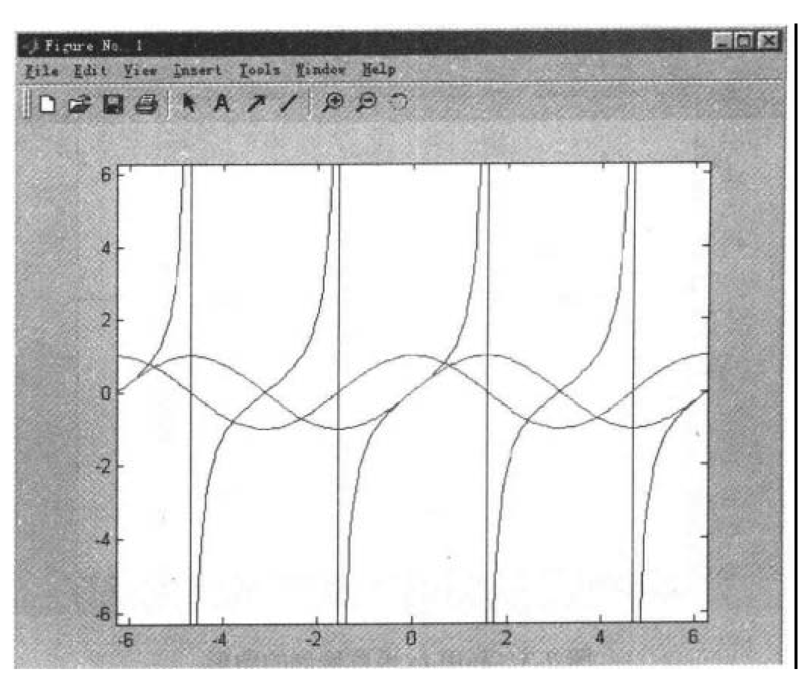



图 7-5　例 7-3 绘制的图形

可以在图 7-5 所示的图形中，单击给图形添加文本说明的A按钮，在图形上分别标注上 tan(x)、sin(x)和 cos(x)，如图 7-6 所示。

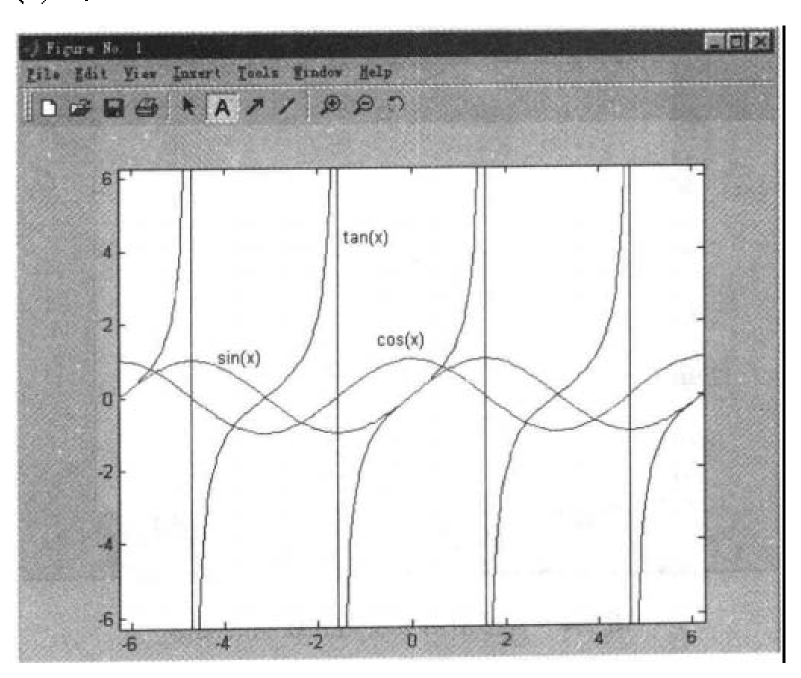



图 7-6　经过添加文本说明后的图形

【例 7-4】

试绘制函数 $\sin\frac{1}{x}$ 在区间 [0.01,0.1]上的图形，要求相对误差为 0.001。可以执行以下命令：

```
fplot('sin(1/ x)', [0.01 0.1],1e-3)
```

以上命令的执行结果如图 7-7 所示。

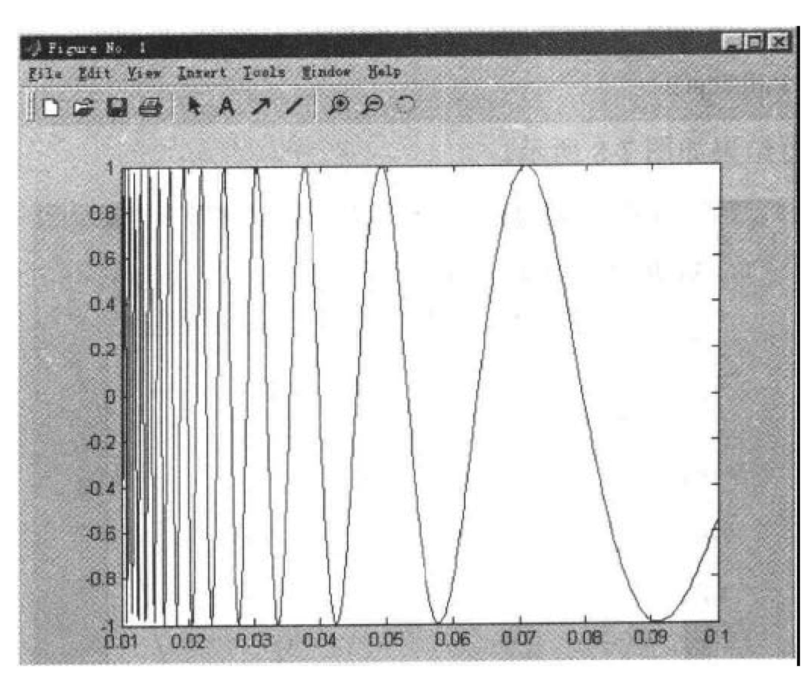


图 7-7 函数 sin(1/x)在区间[0.01,0.1]上的图形

## 7.1.2 ezplot 命令

ezplot 是 Easy to use function plotter 之意，即借助于函数绘图器方便地绘制函数的图形。在前一节中介绍的 fplot 命令只能绘制一元函数的图形。而通过 ezplot 命令却可以绘制二元函数的图形。

对显式函数 $f=f(x)$来说：

- ezplot(f )：在默认区间 $x\in(-2\pi,2\pi)$上绘制函数 $f=f(x)$的图形。
- ezplot(f,[a b])：在区间 $x\in(a,b)$上绘制函数 $f=f(x)$的图形。

对隐式函数 $f=f(x,y)$来说：

- ezplot(f )：在默认区间 $x\in(-2\pi,2\pi)$，$y\in(-2\pi,2\pi)$上绘制函数 $f(x,y)=0$ 的图形。
- ezplot(f,[xmin,xmax,ymin,ymax])：在区间 $x\in$(xmin xmax)、$y\in$(ymin ymax)上绘制函数 $f(x,y)=0$ 的图形。
- ezplot(f,[a,b])：在区间 $x\in(a,b)$、$y\in(a,b)$上绘制函数 $f(x,y)=0$ 的图形。如果函数不是 $x,y$ 的函数，比如说是 $u,v$ 的函数，那么默认绘图区间是按照变量的字母顺序分配的，例如：ezplot ('u^2 - v^2 - 1',[-3,2,-2,3])就是在区间 $u\in(-3,2)$、$v\in(-2,3)$上绘制 $u^2-v^2-1=0$ 的图形。
- ezplot(x,y)：在默认的区间 $t\in(0,2\pi)$上绘制参数函数 $x=x(t)$，$y=y(t)$的图形。
- ezplot(x,y,[tmin,tmax])：在区间 $t\in$(tmin tmax)上绘制参数函数 $x=x(t)$，$y=y(t)$的图形。

以下几条命令都是在给定的绘图区间内，在指定的图形窗口 fig 中绘制函数的图形：

```
ezplot(f,[a,b],fig)
ezplot(f,[xmin,xmax,ymin,ymax],fig)
ezplot(x,y,[tmin,tmax],fig)
```

【例 7-5】

（1）绘制函数 $f(x)=x^2\sin x-1$ 在区间(0,2π)上的图形。可以执行以下命令：

```
ezplot('x^2*sin(x)-1',[0,2*pi])
```

以上命令的执行结果如图 7-8 所示。

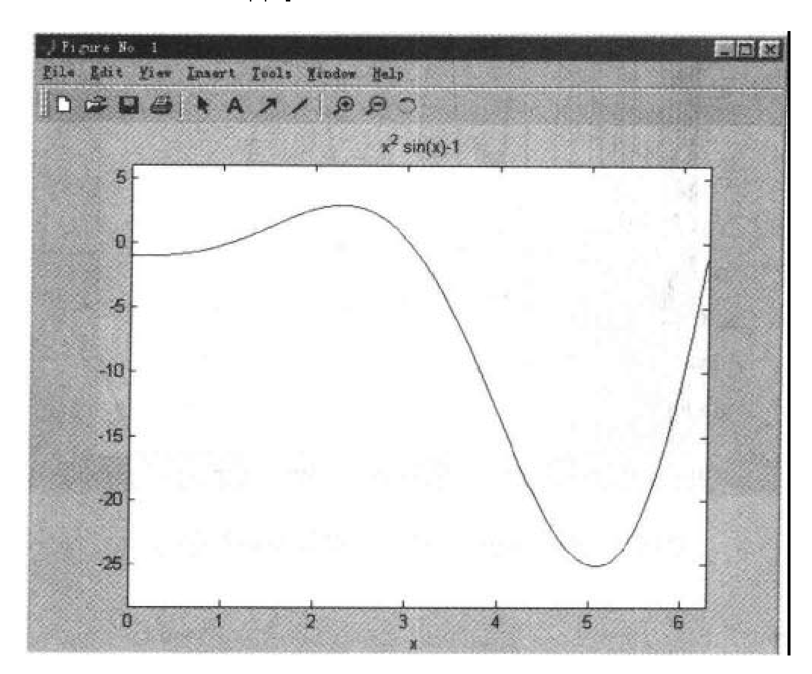



图 7-8　函数 $f(x)=x^2\sin x-1$ 的图形

（2）在区间 $u\in(-3,2)$、$v\in(-2,3)$上绘制 $u^2-v^2-1=0$ 的图形。可以执行以下命令：

```
ezplot ('u^2 - v^2 - 1',[-3,2,-2,3])
```

以上命令的执行结果如图 7-9 所示。

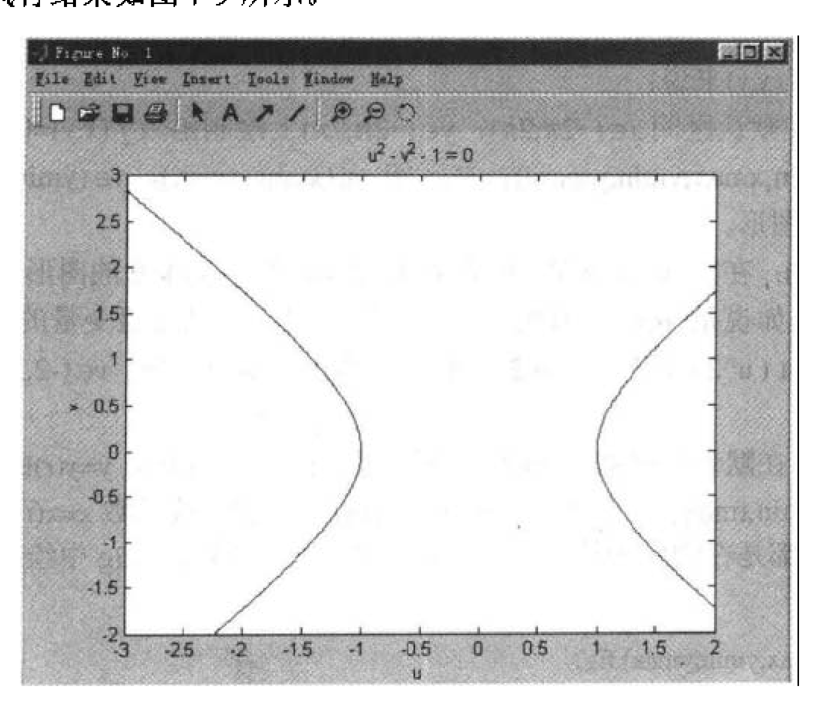



图 7-9　函数 $u^2-v^2-1=0$ 的图形

有时用 ezplot 命令绘制参数式曲线也非常方便。

【例 7-6】

绘制参数式方程：$\begin{cases} x=\sin(3t)\cos(t) \\ y=\sin(3t)\sin(t) \end{cases}$在区间[0,π]上的图形。可以执行以下命令：

```
ezplot('sin(3*t)*cos(t)','sin(3*t)*sin(t)',[0,pi])
```

以上命令的执行结果如图 7-10 所示。

☞ 提示：从例 7-5 和例 7-6 可以看出，与 plot 命令以及 fplot 命令相比，ezplot 命令的一个最大优点就是它不仅可在绘制的图形上自动标注横、纵坐标轴，而且可以自动进行图题标注。

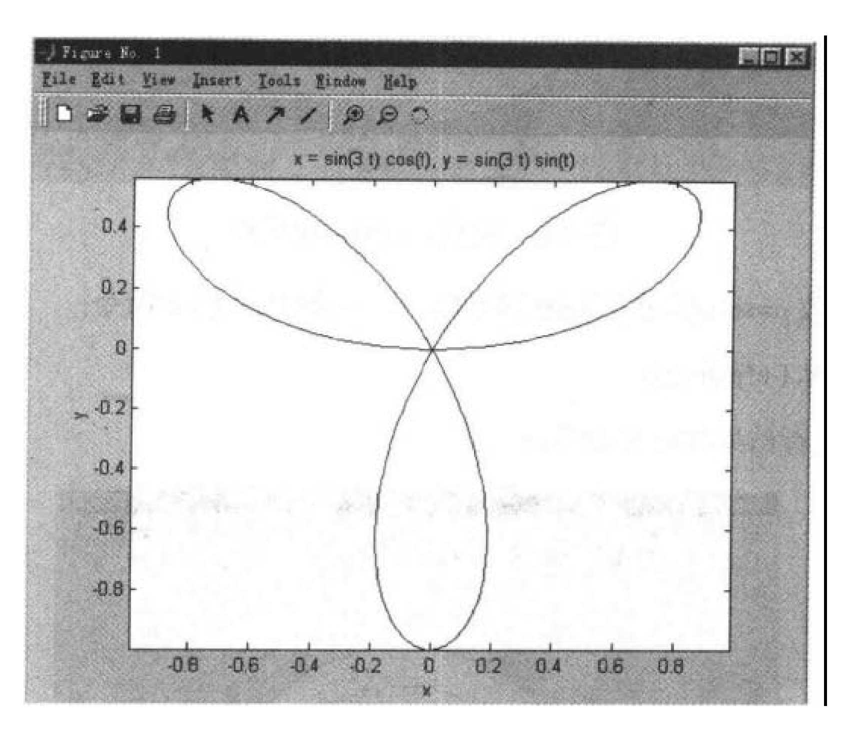



图 7-10 例 7-6 的执行结果

### 7.1.3 ezpolar 命令

有些函数在极坐标系中表达是非常方便的，MATLAB 6.x 提供的 ezpolar 命令（意为：Easy to use polar coordinate plotter）可以实现在极坐标系中绘制符号函数的图形。其调用格式为：

- ezpolar(f)：在默认区间 theta∈(0,2π)上绘制函数 rho=f(theta)的图形，其中 theta 为极角，rho 为极径。
- ezpolar(f,[a,b])：在区间 theta∈($a$,$b$)上绘制函数 rho=f(theta)的图形，其中 theta 为极角，rho 为极径。

【例 7-7】

（1）绘制函数 $\rho$=1+cos($t$)的图形，可以执行以下命令：

```
ezpolar('1 + cos(t)')
```

以上命令的执行结果如图 7-11 所示。

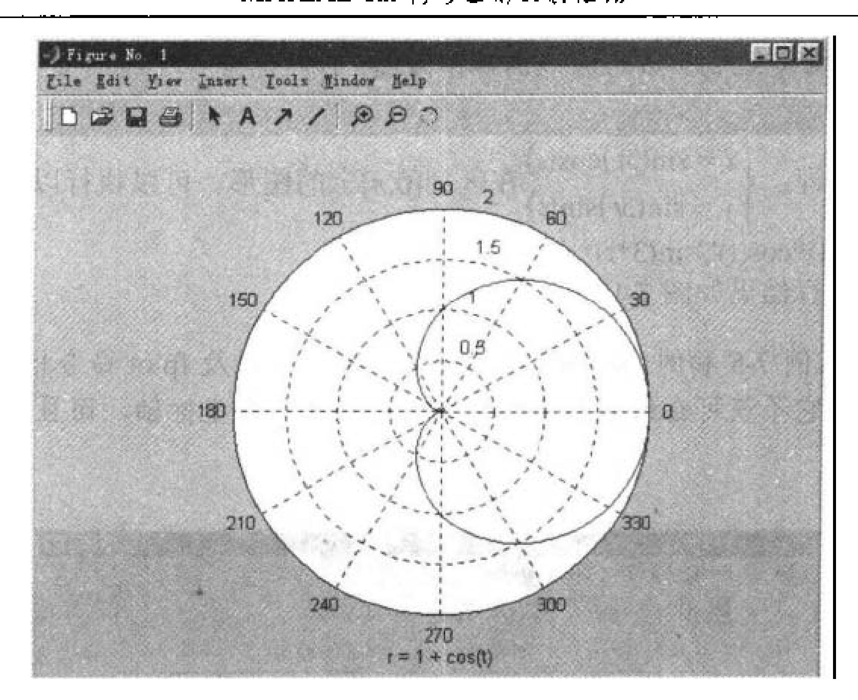



图 7-11 函数 $\rho$=1+cos($t$)的图形

（2）绘制函数 $\rho$=sin($t$)/$t$ 在[-6π,6π]上的图形，可以执行以下命令：

```
ezpolar('sin(t)/t', [-6*pi,6*pi])
```

以上命令的执行结果如图 7-12 所示。

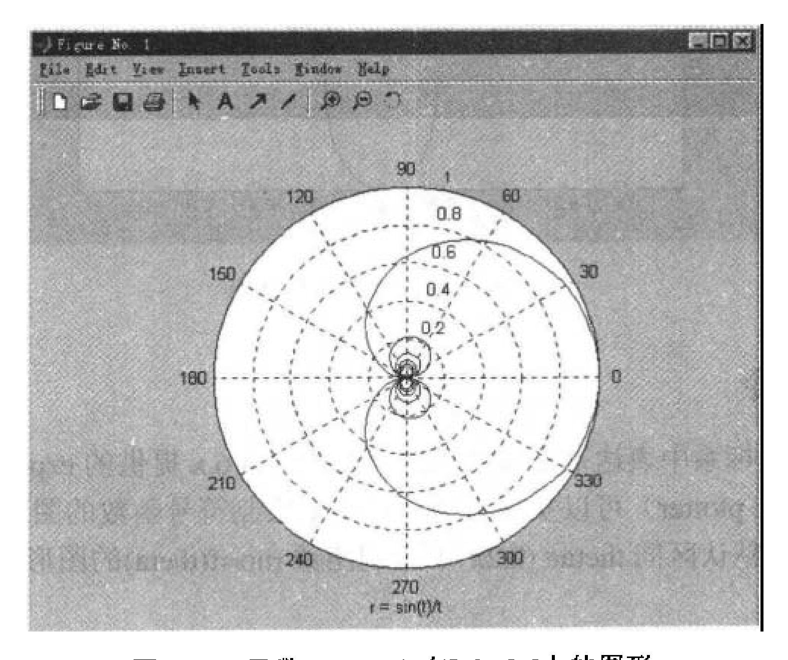



图 7-12 函数 $\rho$=sin($t$)/$t$ 在[-6π,6π]上的图形

（3）ezpolar 命令还可以绘制表达形式较为复杂的极坐标图形。执行以下命令：

```
r = '100/(100+(t-1/2*pi)^8)*(2-sin(7*t)-1/2*cos(30*t))';
ezpolar(r,[-pi/2,3*pi/2])
```

以上命令的执行结果如图 7-13 所示。

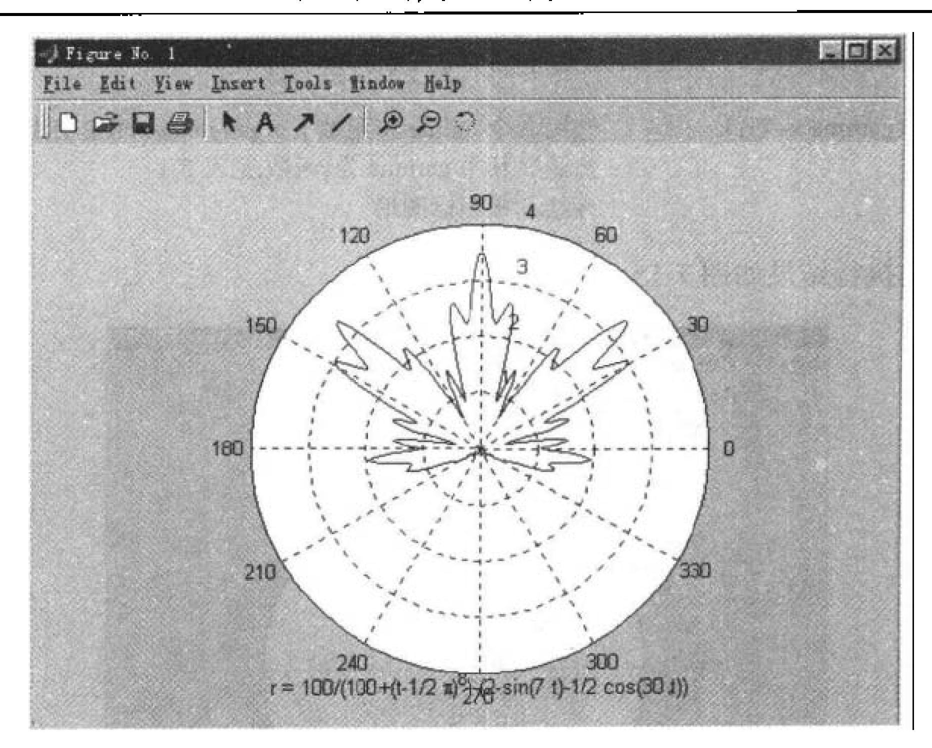



图 7-13 用 ezpolar 命令绘制的复杂极坐标图形

（4）像 plot、fplot 命令一样，也可以用函数句柄的形式绘制极坐标图形。执行以下命令：

```
ezpolar(@cos,[0,pi])
```

以上命令的执行结果如图 7-14 所示。

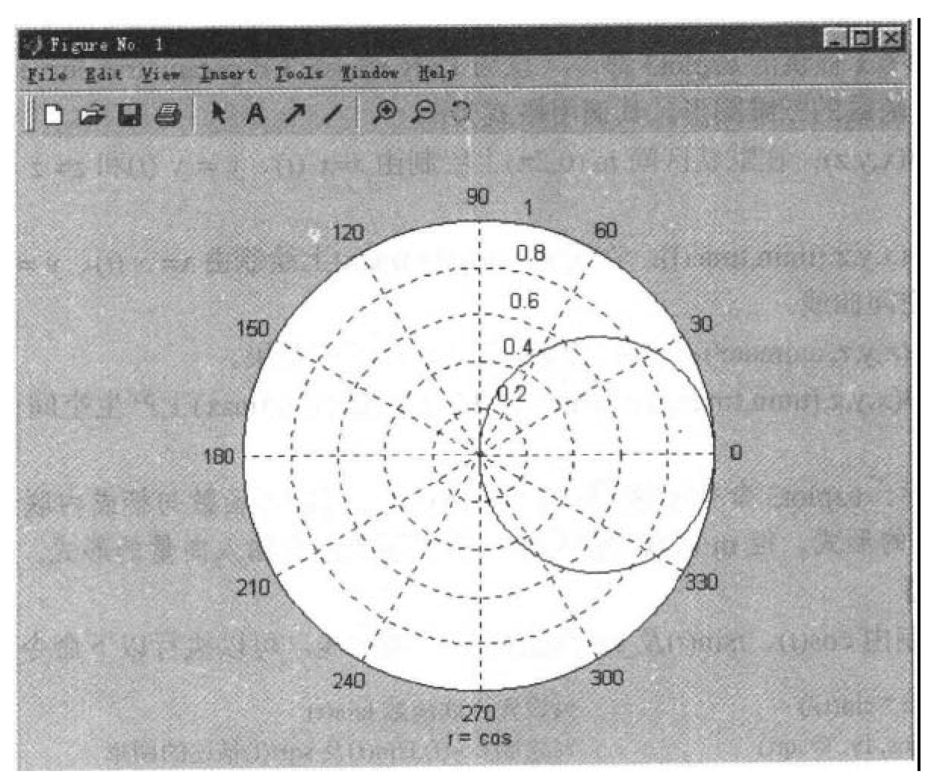



图 7-14 利用函数句柄绘制的极坐标图形

（5）使用内联函数的办法绘制极坐标图形。执行以下命令：

```
h = inline('log(gamma(x+1))');        %用命令 inline 将函数 log(gamma(x+1))说明为内联
                                      %函数其中 gamma 为γ函数,见表 5-1
ezpolar(h)                            %绘制极坐标图形
```

以上命令的执行结果如图 7-15 所示。

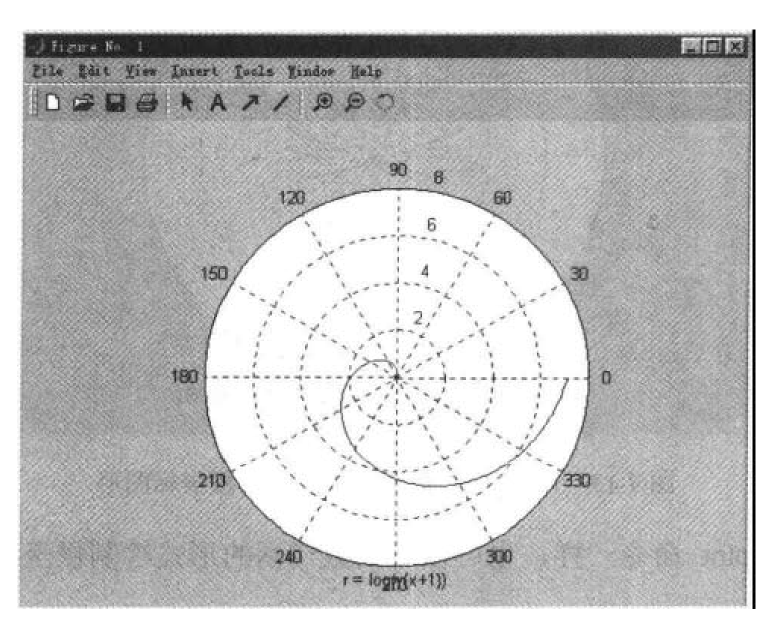



图 7-15　使用内联命令绘制的极坐标图形

### 7.1.4　ezplot3 命令

MATLAB 6.x 提供的 ezplot3 命令（意为 Easy to use 3-d parametric curve plotter）可以方便地绘制符号函数的三维图形。其调用格式为：

- ezplot3(x,y,z)：在默认区间 $t\in(0,2\pi)$上绘制由 $x=x(t)$、$y=y(t)$和 $z=z(t)$描述的空间曲线。
- ezplot3(x,y,z,[tmin,tmax])：在区间 $t\in$(tmin,tmax)上绘制由 $x=x(t)$、$y=y(t)$和 $z=z(t)$描述的空间曲线。
- ezplot3(x,y,z,'animate')：产生空间曲线的动画追踪效果。
- ezplot3(x,y,z,[tmin,tmax],'animate')：在区间 $t\in$(tmin,tmax)上产生空间曲线的动画绘制效果。

☞ 提示：ezplot3 命令中使用的函数 $x$，$y$ 和 $z$ 可以是函数句柄或内联函数的形式，也可以是表达式的形式。但 m 函数和内联函数必须写成接受输入向量的形式。

【例 7-8】

（1）绘制由 cos($t$)、$t$sin($t$)及 sqrt($t$)描述的空间曲线，可以执行以下命令：

```
fy = inline('t * sin(t)')             %设置内联函数 tsin(t)
ezplot3(@cos, fy, @sqrt)              %绘制 cos(t),tsin(t)及 sqrt(t)描述的图形
```

以上命令的执行结果如图 7-16 所示。

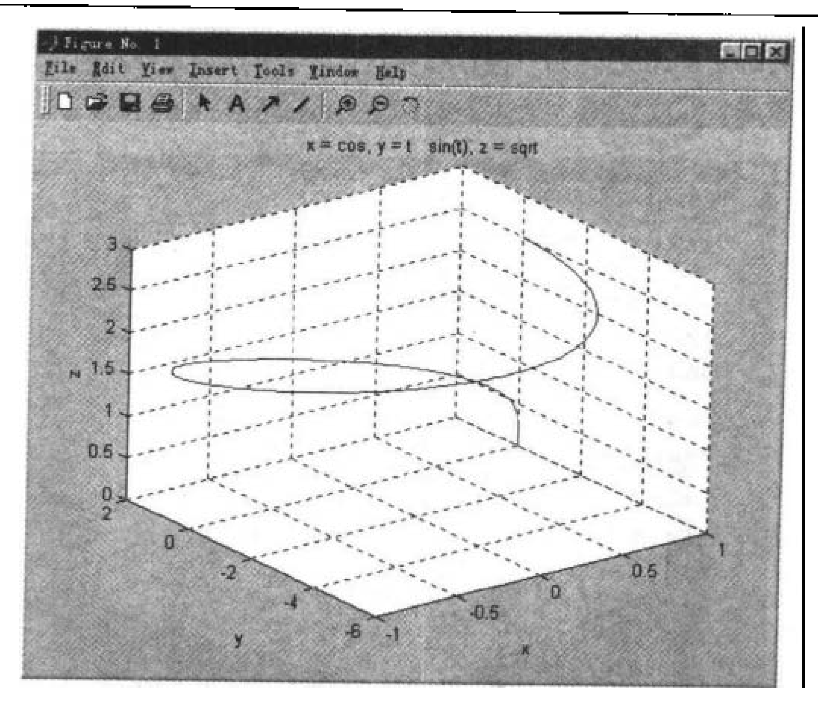



图 7-16　由 cos(t)、tsin(t)及 sqrt(t)描述的空间曲线

（2）绘制由参数表达式 $x$=cos($t$),$y$=$t$sin($t$),$z$=sqrt($t$)描述的图形，可以执行以下命令：

```
ezplot3('cos(t)', 't * sin(t)', 'sqrt(t)', [0,6*pi])
%在区间(0,6π)上绘制由 x=cos(t),y=tsin(t),z=sqrt(t)描述的图形
```

以上命令的执行结果如图 7-17 所示。

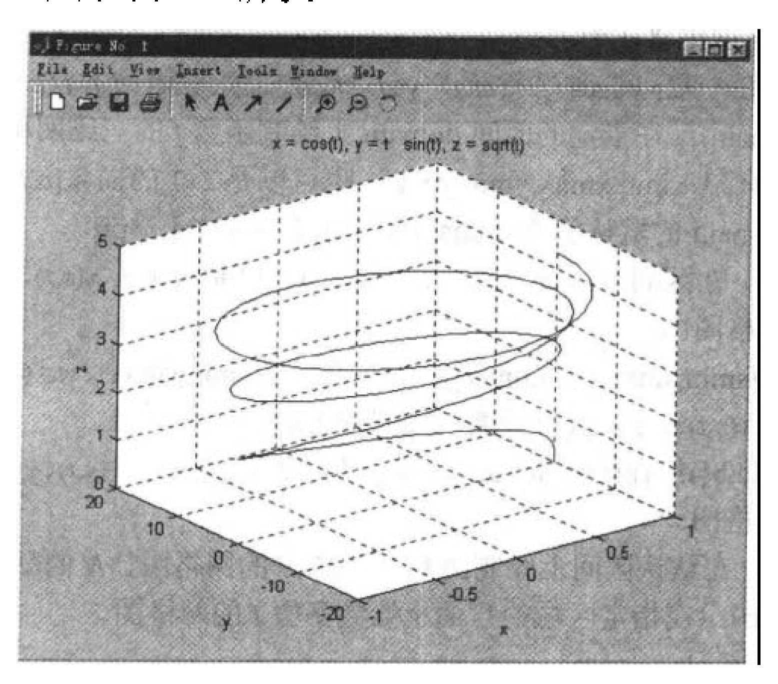



图 7-17　由 x=cos(t),y=tsin(t),z=sqrt(t)描述的图形

（3）执行以下命令：

```
fy = inline('t .* sin(t)')                %设置内联函数 tsin(t)
ezplot3(@cos, fy, @sqrt, 'animate')       %用函数句柄以及内联函数的形式动态绘制三维图形
```

图 7-18 是以上命令的最终执行结果。执行命令过程中，可以看到小红球由低向高迅速移动，表达了动态效果。

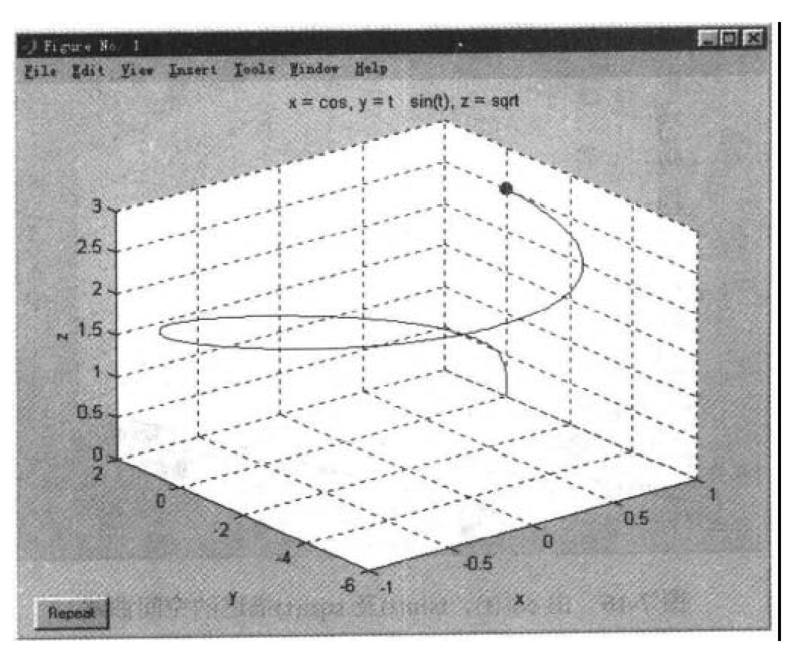



图 7-18 用 ezplot3 绘制的动画

## 7.1.5 ezmesh 命令

MATLAB 6.x 提供的 ezmesh 命令（意为 Easy to use 3-D mesh plotter）可以绘制符号函数的三维网格图，其调用格式为：

- ezmesh(f)：在默认区间 $x\in(-2\pi,2\pi)$，$y\in(-2\pi,2\pi)$上绘制函数 $f=f(x,y)$的网格图。
- ezmesh(f,domain)：在指定的区间 domain 上绘制函数 $f$ 的三维网格图。其中 domain 可以是 4×1 的向量[xmin,xmax,ymin,ymax]，也可以是 2×1 的向量[$a$,$b$]（此时区间为 $a<x<b$，$a<y<b$），domain 的默认值为 domain=[-2*pi,2*pi,-2*pi,2*pi]。
- emesh(x,y,z)：在区间 $s\in(-2\pi,2\pi)$，$t\in(-2\pi,2\pi)$上绘制由 $x = x(s,t)$、$y=y(s,t)$及 $z=z(s,t)$函数描述的网格图。
- emesh(x,y,z,[smin,smax,tmin,tmax])：在区间 $s\in$(smin,smax)，$t\in$(tmin,tmax)上绘制由 $x=x(s,t)$、$y=y(s,t)$及 $z=z(s,t)$函数描述的网格图。
- emesh(x,y,z,[a,b])：在区间 $s\in(a,b)$，$t\in(a,b)$上绘制由 $x=x(s,t)$、$y=y(s,t)$及 $z=z(s,t)$函数描述的网格图。
- emesh(...,N)：在默认区间上绘制函数 $f$ 的 $N\times N$ 的网格图，$N$ 的默认值为 60。
- emesh(...,'circ')：在指定区域的盘面上绘制函数 $f$ 的网格图。

注意：在调用格式 ezmesh(f,domain)中，若自变量不是 $x,y$，比如说自变量是 $u,v$，那么 umin、umax、vmin 及 vmax 是按照字母的顺序排列的。因此，emesh('u^2-v^3',[0,1,3,6]) 表示在区间 $0<u<1,3<v<6$ 上绘制 u^2-v^3=0 的三维网格图。

提示：ezmesh 命令中使用的函数 $x$、$y$ 和 $z$ 可以是函数句柄或内联函数的形式，也可以是表达式的形式。

【例 7-9】

（1）绘制由 ezmesh(f,domain)格式得到的三维网格图，可以执行以下命令：

```
f = ['3*(1-x)^2*exp(-(x^2) - (y+1)^2)- 10*(x/5 - x^3 - y^5)*exp(-x^2-y^2)' ...
      '- 1/3*exp(-(x+1)^2 - y^2)'];              %指定函数 f 的表达式
ezmesh(f,[-pi,pi])                               %在区间(-π,π)上绘制函数 f 的三维网格图
```

以上命令的执行结果如图 7-19 所示。

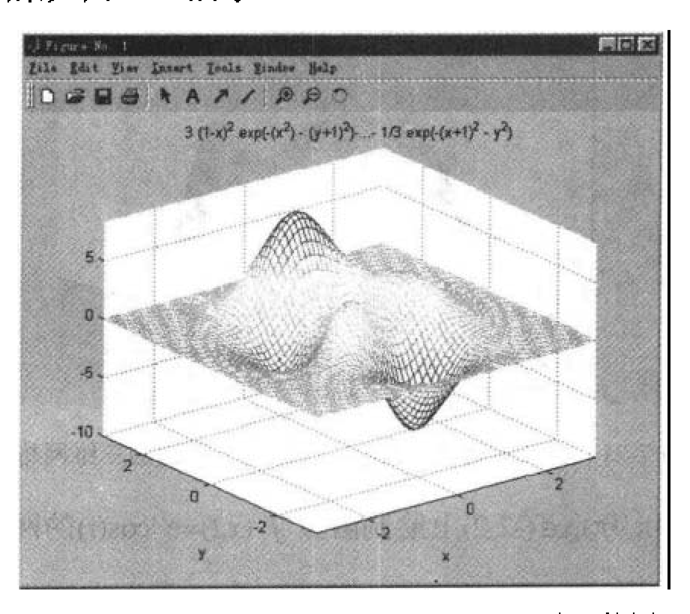



图 7-19　由 ezmesh(f,domain)格式得到的三维网格图

（2）绘制函数 $x \cdot \exp(-x^2-y^2)$的网格图，可以执行命令：

```
ezmesh('x*exp(-x^2 - y^2)')              %绘制函数 x*exp(-x^2 - y^2)的网格图
```

以上命令的执行结果见图 7-20 所示。

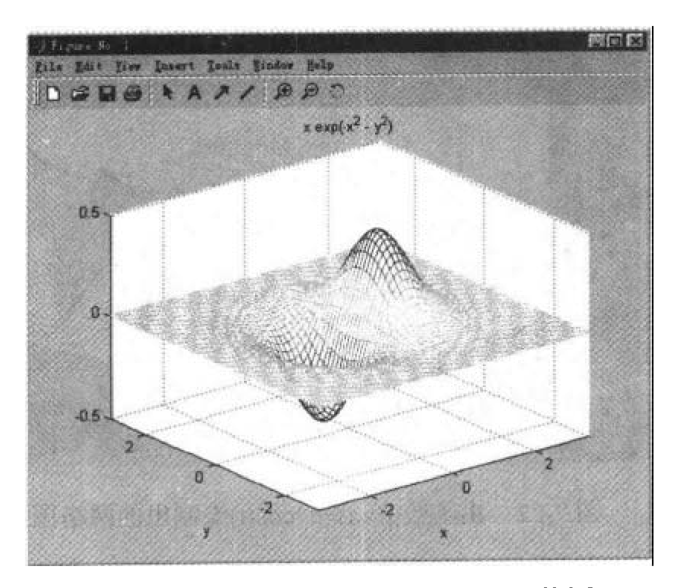



图 7-20　由函数 $x \cdot \exp(-x^2-y^2)$绘制出的网格图

（3）通过调用格式 emesh(...,'circ') 绘制出三维网格图，可以执行命令：

```
ezmesh('x*y','circ')                %在默认的盘面上绘制函数 xy=0 的三维网格图
```

以上命令的执行结果如图 7-21 所示。

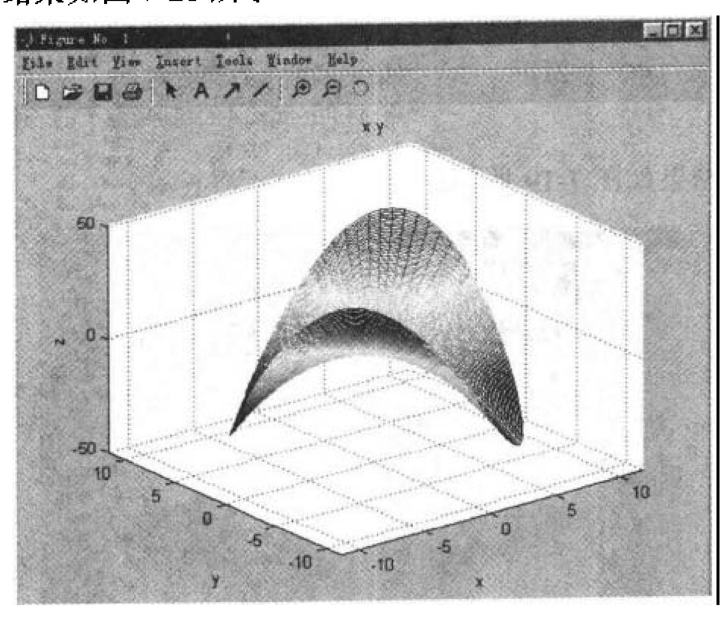



图 7-21 由调用格式 emesh(...,'circ') 绘制出的三维网格图

（4）在区间 $x\in(-4\pi,4\pi),y\in(-2,2)$ 上绘制函数 $f(x,t)=e^{-x}\cos(t)$的网格图。可以执行以下命令：

```
ezmesh('exp(-x)*cos(t)',[-4*pi,4*pi,-2,2])
```

以上命令的执行结果如图 7-22 所示。

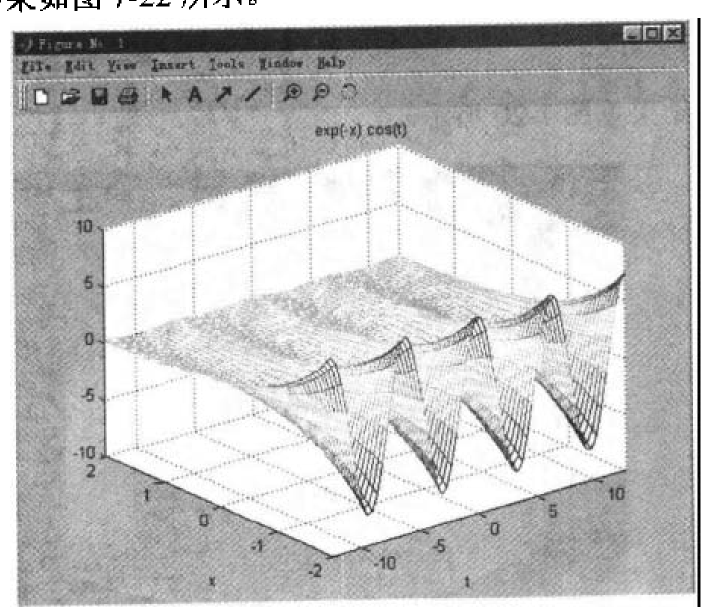



图 7-22 由函数 $f(x,t)=e^{-x}\cos(t)$绘制出的网格图

（5）通过内联函数得到的三维网格图，可以执行以下命令：

```
h = inline('x*y - x');          %设定内联函数
ezmesh(h)                       %绘制内联函数的三维网格图
```

以上命令的执行结果如图 7-23 所示。

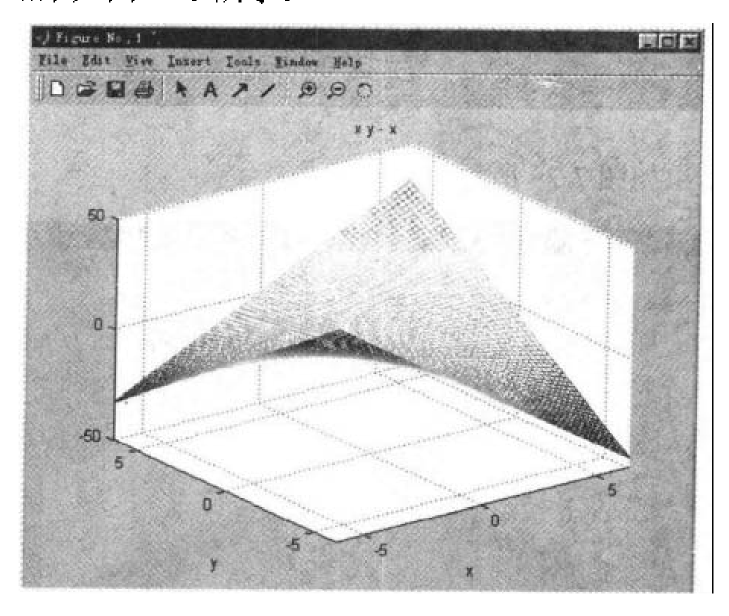


图 7-23 由内联函数得到的三维网格图

（6）执行以下命令：

```
ezmesh('(s-sin(s))*cos(t)','(1-cos(s))*sin(t)','s',[-2*pi,2*pi])
```

函数 $x$=$s$-sin($s$)*cos($t$),$y$=(1-cos($s$))*sin($t$),$z$=$s$ 所对应的三维网格图的执行结果如图 7-24 所示。

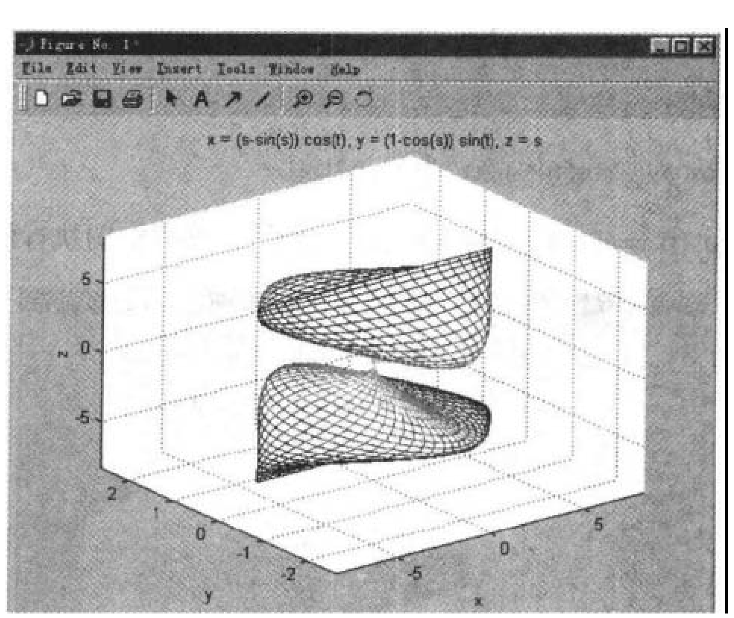



图 7-24 函数 x=s-sin(s)*cos(t),y=(1-cos(s))*sin(t),z=s 对应的三维网格图

☞ 提示：MATLAB 6.x 还提供了 ezmeshc 命令，它的调用方法和 ezmesh 命令完全相同，只是所绘出的图形效果不同，ezmeshc 命令在绘制三维网格图的同时还绘制出等值线图。

【例 7-10】

（1）通过 ezmeshc(f,domain)格式得到的带有等值线的三维网格图，可执行以下命令：

```
f = ['3*(1-x)^2*exp(-(x^2) - (y+1)^2)- 10*(x/5 - x^3 - y^5)*exp(-x^2-y^2)' ...
    '- 1/3*exp(-(x+1)^2 - y^2)'];          %指定函数 f 的表达式
ezmeshc(f,[-pi,pi])                         %在区间(-π,π)上绘制函数 f 带有等值线的三维
                                            %网格图
```

以上命令的执行结果如图 7-25 所示。

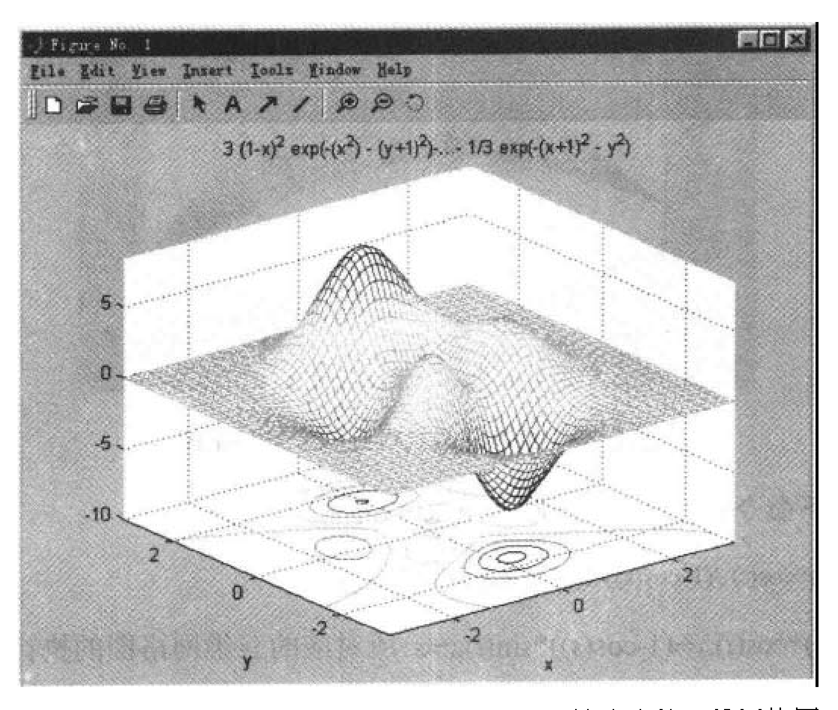



图 7-25 由 ezmeshc(f,domain)格式得到的带有等值线的三维网格图

（2）执行以下命令：

```
ezmeshc('(s-sin(s))*cos(t)','(1-cos(s))*sin(t)','s',[-2*pi,2*pi])
```

函数 $x=s\text{-}\sin(s)*\cos(t),y=(1\text{-}\cos(s))*\sin(t),z=s$ 对应的三维网格图执行结果，如图 7-26 所示。

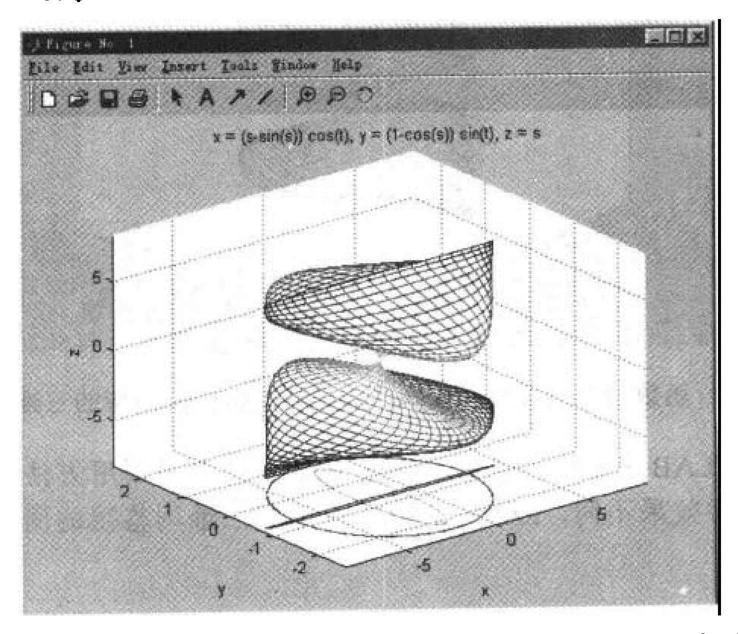



图 7-26 函数 $x=s\text{-}\sin(s)*\cos(t),y=(1\text{-}\cos(s))*\sin(t),z=s$ 对应的三维网格图

### 7.1.6 ezcontour 命令

MATLAB 6.x 提供的 ezcontour 命令（意为 Easy to use contour plotter）可以方便地绘制符号函数的等值线图，其调用格式为：

ezcontour(f)：在默认区间∈(-2π,2π),y∈(-2π,2π)上绘制函数 $f=f(x,y)$的等值线图。

- ezcontour(f,domain)：在指定的区间 domain 上绘制函数 $f$ 的等值线图。其中 domain 可以是 4×1 的向量[xmin,xmax,ymin,ymax]，也可以是 2×1 的向量[a,b]（此时区间为 $a<x<b$, $a<y<b$），domain 的默认值为 domain=[-2*pi,2*pi,-2*pi,2*pi]。
- ezcontour(...,N)：在默认区间上绘制函数 $f$ 的 $N\times N$ 个网格的等值线图，$N$ 的默认值为 60。

注意：在调用格式 ezcontour(f,domain)中，若自变量不是 $x,y$，比如说自变量是 $u,v$，那么 umin、umax、vmin 及 vmax 是按照字母的顺序排列的。因此，emesh('u^2- v^3',[0,1,3,6])表示在区间 $0<u<1, 3<v<6$ 上绘制 $u$^2-$v$^3=0 的等值线图。

【例 7-11】

（1）通过 ezcontour(f,domain)格式得到等值线图，可以执行以下命令：

```
f = ['3*(1-x)^2*exp(-(x^2) - (y+1)^2)- 10*(x/5 - x^3 - y^5)*exp(-x^2-y^2)' ...
        '- 1/3*exp(-(x+1)^2 - y^2)'];          %指定函数 f 的表达式
ezcontour(f,[-pi,pi])                        %在区间(-π,π)上绘制函数 f 的等值线图
```

以上命令的执行结果如图 7-27 所示。

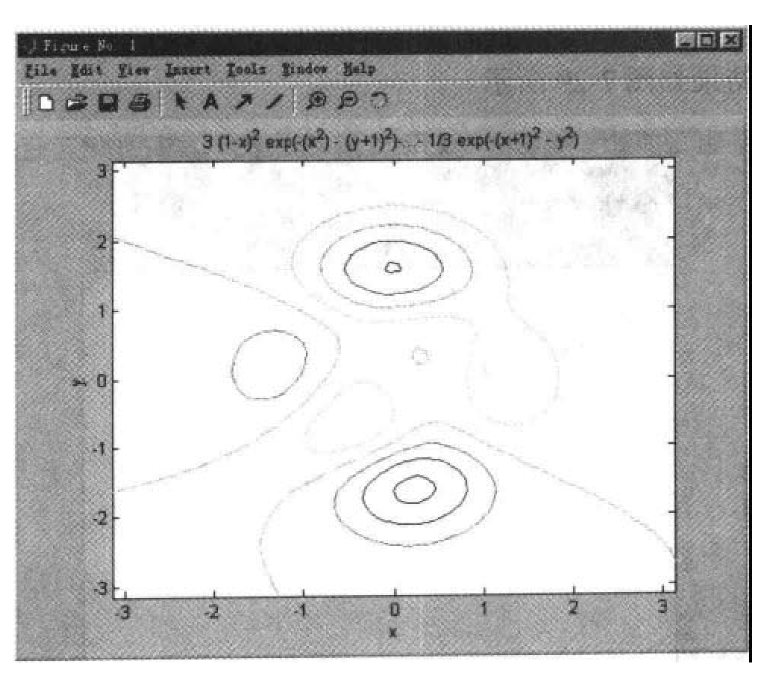



图 7-27 由 ezcontour(f,domain)格式得到的等值线图

（2）通过 ezcontour(...,N)格式得到等值线图，可以执行以下命令：

```
ezcontour('-3*z/(1 + t^2 - z^2)',[-4,4],120)
%在区间-4<t<4, -4<z<4 上绘制函数-3*z/(1 + t^2-z^2)由 120×120 个网格所决定等值线图
```

以上命令的执行结果如图 7-28 所示。

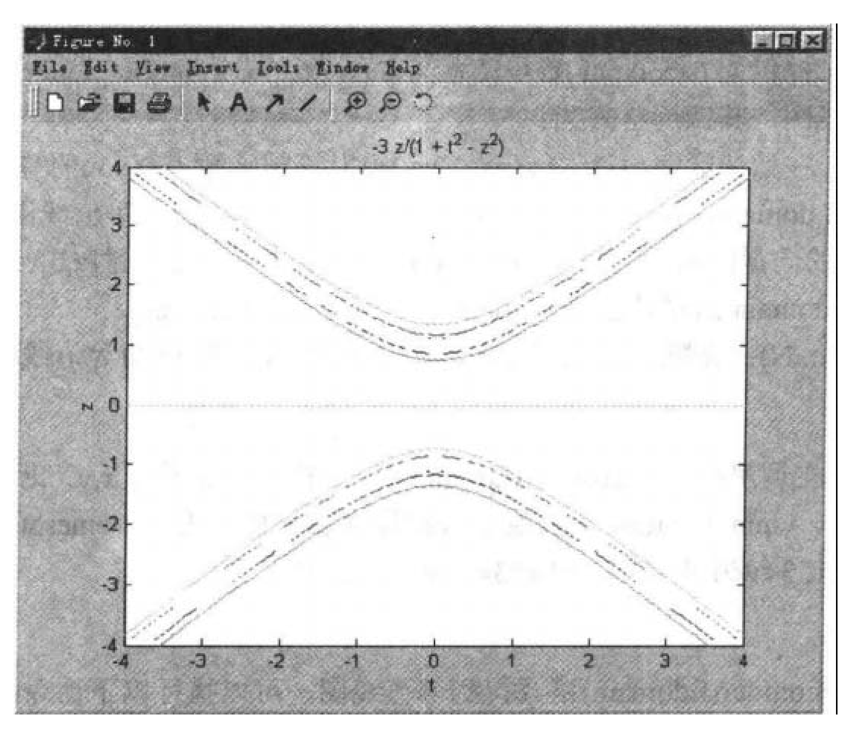



图 7-28 由 ezcontour(...,N)格式得到的等值线图

（3）通过内联函数得到等值线图，可以执行以下命令：

```
h = inline('x*y - x');          %设定内联函数
ezcontour(h)                    %绘制内联函数的等值线图
```

以上命令的执行结果如图 7-29 所示。

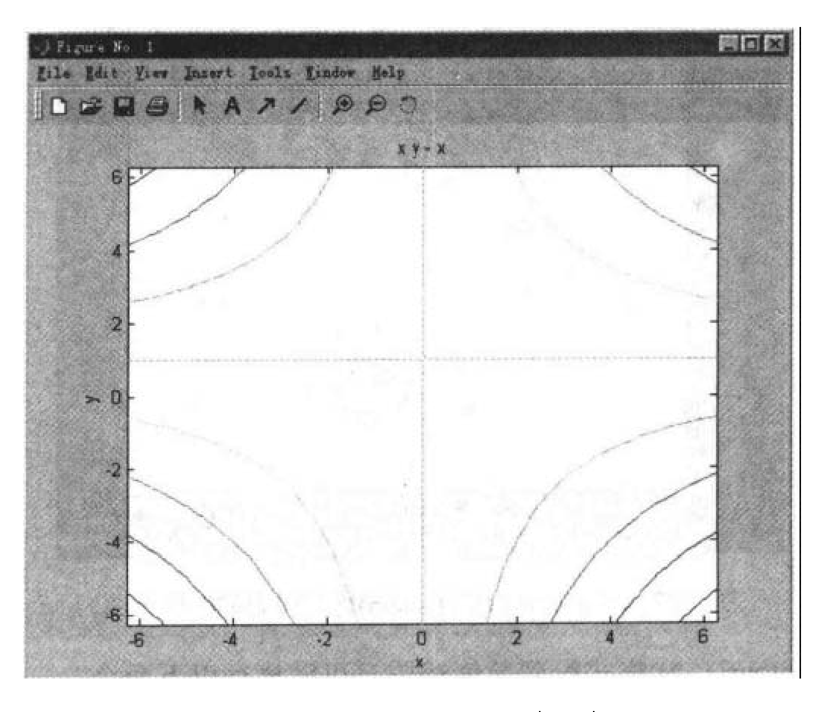



图 7-29 由内联函数得到的等值线图

☞ 提示：MATLAB 6.x 还提供了 ezcontourf 命令（意为 Easy to use filled contour plotter），它的调用方法和 ezcontour 命令完全相同，只是所绘制出的图形效果有所差别，ezcontourf 命令绘制的图形是经过填充的等值线图。

【例 7-12】

（1）通过 ezcontourf(f,domain)格式得到经过填充的等值线图，可以执行以下命令：

```
f = ['3*(1-x)^2*exp(-(x^2) - (y+1)^2)- 10*(x/5 - x^3 - y^5)*exp(-x^2-y^2)' ...
        '- 1/3*exp(-(x+1)^2 - y^2)'];          %指定函数 f 的表达式
ezcontourf(f,[-pi,pi])                          %在区间(-π,π)上绘制函数 f 经过填充的等值线图
```

以上命令的执行结果如图 7-30 所示。

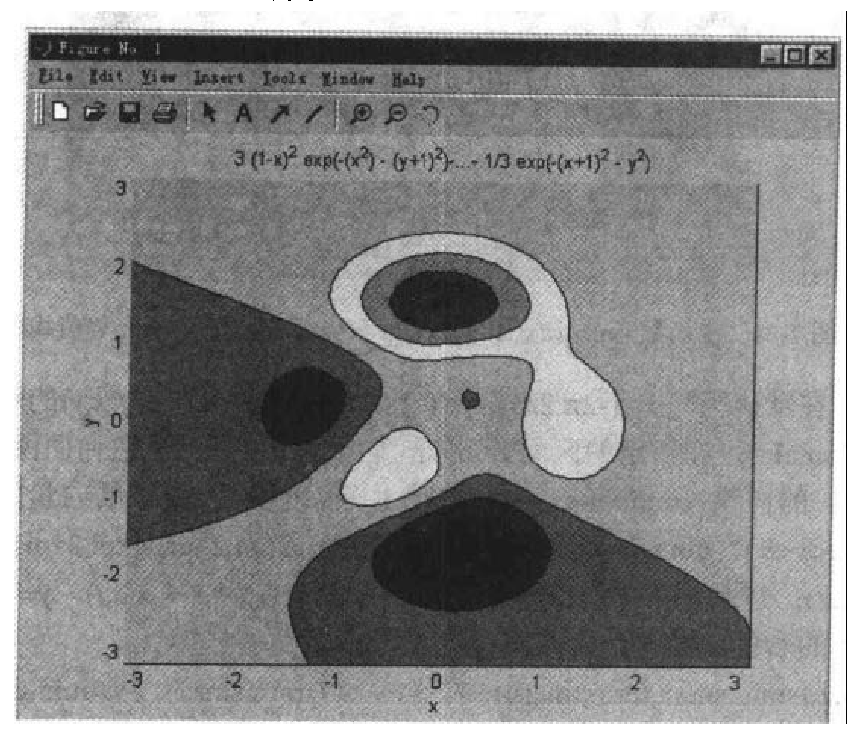



图 7-30 由 ezcontourf(f,domain)格式得到的经过填充的等值线图

（2）绘制函数 sin(sqrt($x$^2+$y$^2))/sqrt($x$^2+$y$^2)的填充等值线图，可以执行以下命令：

```
ezcontourf('sin(sqrt(x^2+y^2))/sqrt(x^2+y^2)',[-6*pi,6*pi])
                    % 在 x∈(-6π,6π)，y∈(-6π,6π)上绘制函数 sin(sqrt(x^2+y^2))/sqrt(x^2+y^2)
                    % 经过填充处理的等值线图
```

以上命令执行结果如图 7-31 所示。

☞ 提示：ezcontourf 命令绘制的等值线图是经过填充处理的，显然，在视觉效果上要比用 ezcontour 命令绘制的等值线图形美观一些。

## 7.1.7 ezsurf 命令

MATLAB 6.x 提供的 ezsurf 命令（意为 Easy to use 3-D colored surface plotter）可以绘制符号函数的彩色曲面图，其调用格式为：

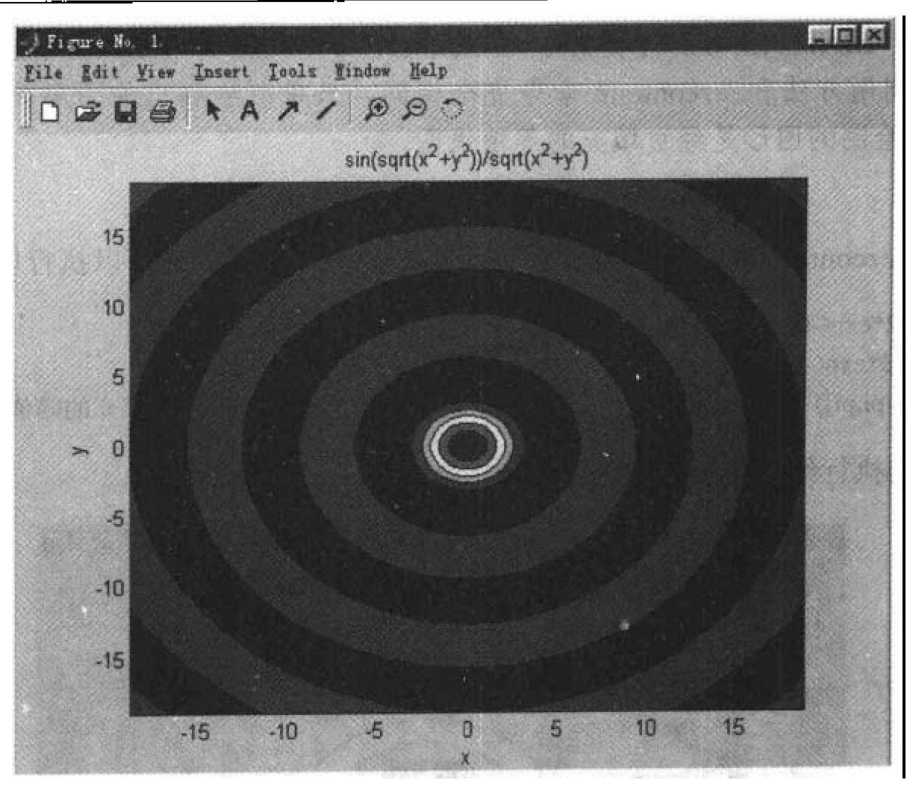



图 7-31 由函数 sin(sqrt(x^2+y^2))/sqrt(x^2+y^2)绘制的填充等值线图

- ezsurf(f)：在默认区间 $x\in(-2\pi,2\pi)$，$y\in(-2\pi,2\pi)$上绘制函数 $f=f(x,y)$的彩色曲面图。
- ezsurf(f,domain)：在指定的区间 domain 上绘制函数 $f$ 的彩色曲面图。其中 domain 可以是 4×1 的向量[xmin,xmax,ymin,ymax]，也可以是 2×1 的向量[a,b]（此时区间为 $a<x<b$，$a<y<b$），domain 的默认值为 domain=[-2*pi,2*pi,-2*pi,2*pi]。
- ezsurf(x,y,z)：在区间 $s\in(-2\pi,2\pi)$，$t\in(-2\pi,2\pi)$上绘制由 $x=x(s,t)$，$y=y(s,t)$及 $z=z(s,t)$函数描述的彩色曲面图。
- ezsurf(x,y,z,[smin,smax,tmin,tmax])：在区间 $s\in$(smin,smax)，$t\in$(tmin,tmax)上绘制由 $x=x(s,t)$，$y=y(s,t)$及 $z=z(s,t)$函数描述的彩色曲面图。
- ezsurf(x,y,z,[a,b])：在区间 $s\in(a,b)$，$t\in(a,b)$上绘制由 $x=x(s,t)$、$y=y(s,t)$及 $z=z(s,t)$函数描述的彩色曲面图。
- ezsurf(...,N)：在默认区间上绘制函数 $f$ 的 $N\times N$ 的彩色曲面图，$N$ 的默认值为 60。
- ezsurf(...,'circ')：在指定区域的盘面上绘制函数 $f$ 的彩色曲面图。

注意：在调用格式 ezmesh(f,domain)中，若自变量不是 $x,y$，比如说自变量是 $u,v$，那么 umin, umax, vmin 及 vmax 是按照字母的顺序排列的。因此，emesh('u^2-v^3',[0,1,3,6])表示在区间 $0<u<1$, $3<v<6$ 上绘制 $u$^2-$v$^3=0 的彩色曲面图。

【例 7-13】

（1）绘制函数 $x$*($y$^2)/($x$^2 + $y$^4)的彩色曲面图，可以执行以下命令：

```
ezsurf('x*(y^2)/(x^2 + y^4)')   %在默认区间上绘制函数 x*(y^2)/(x^2 + y^4)的彩色曲面图
```

以上命令的执行结果如图 7-32 所示。

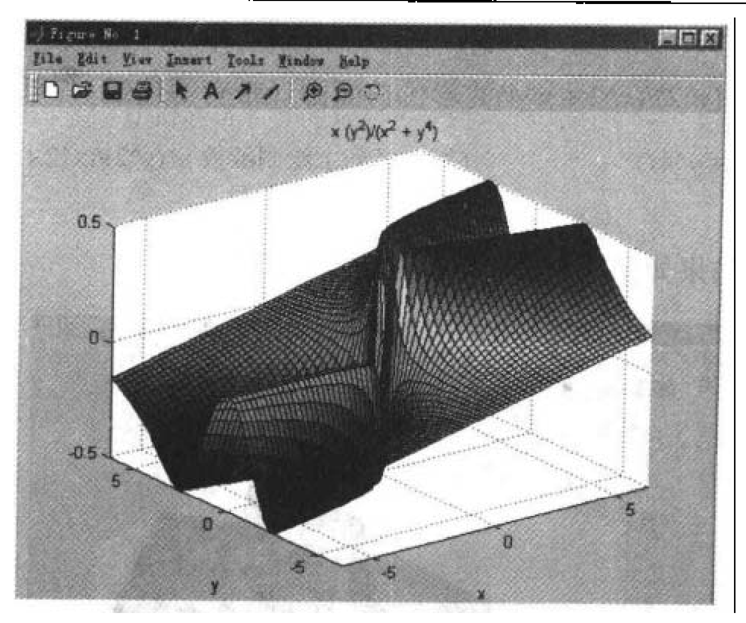



图 7-32　由函数 x*(y^2)/(x^2 + y^4)绘制的彩色曲面图

（2）在指定区间上绘制彩色曲面图，可以执行以下命令：

```
ezsurf('cos(s)*cos(t)','cos(s)*sin(t)','sin(s)',[0,pi/2,0,3*pi/2])
        %在指定区间上绘制函数 x=cos(s)*cos(t),y=cos(s)*sin(t),z=sin(s)的彩色曲面图
```

以上命令的执行结果见图 7-33 所示。

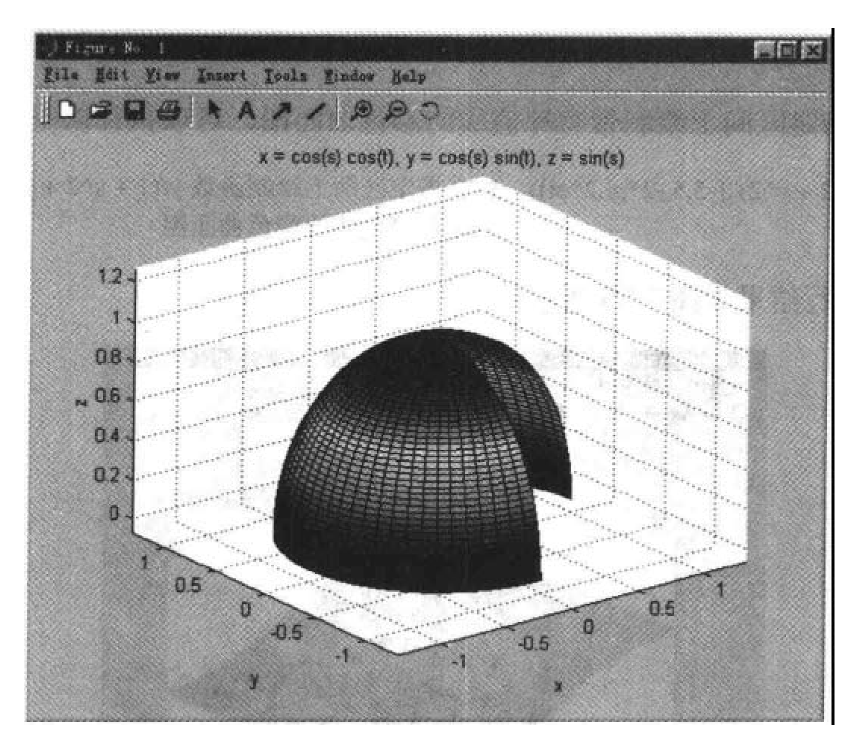



图 7-33　在指定区间上绘制的彩色曲面图

☞ 提示：MATLAB 6.x 还提供了 ezsurfc 命令（意为 Easy to use combination surf/contour plotter），它的调用方法和 ezsurf 命令完全相同，只是所绘出的图形效果有所差别，ezsurfc 命令在绘制曲面图的同时还绘制出等值线图。

【例 7-14】

（1）绘制函数 x*(y^2)/(x^2 + y^4)的彩色曲面图（带有等值线），可以执行以下命令：

```
ezsurfc('x*(y^2)/(x^2 + y^4)')          %在默认区间上绘制函数 x*(y^2)/(x^2 + y^4)带有等
                                         %值线图的彩色曲面图
```

以上命令的执行结果如图 7-34 所示。

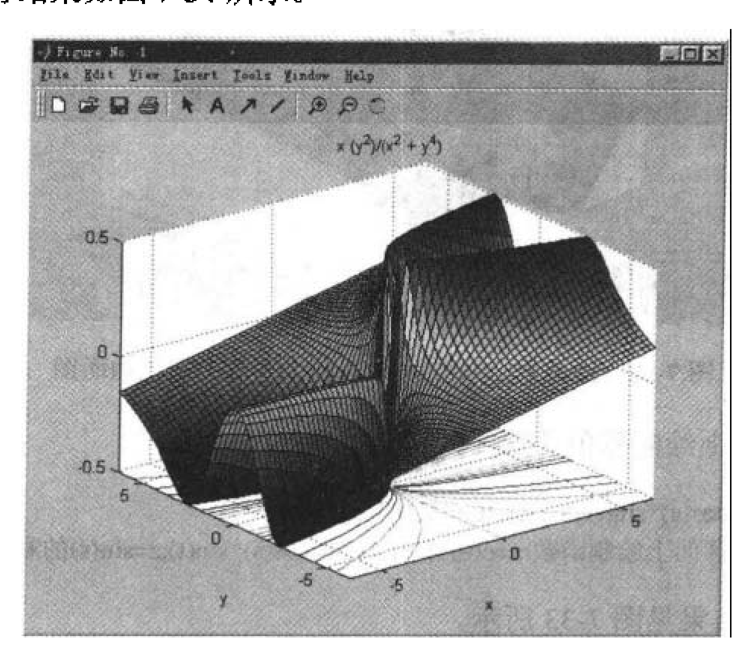



图 7-34　由函数 x*(y^2)/(x^2 + y^4)绘制的彩色曲面图（带有等值线）

（2）通过在指定区间上绘制带有等值线的彩色曲面图，可以执行以下命令：

```
ezsurfc('y/(1 + x^2 + y^2)',[-5,5,-2*pi,2*pi])  %在指定区间上绘制函数 y/(1 + x^2 + y^2)
                                                 %带有等值线的彩色曲面图
```

以上命令的执行结果如图 7-35 所示。

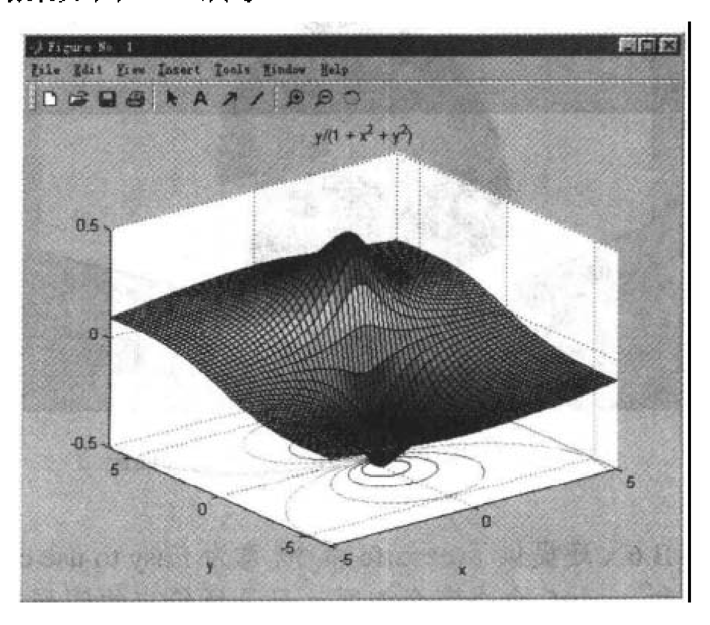



图 7-35　在指定区间上绘制的带有等值线的彩色曲面图

## 7.2 图形的控制与标注

一幅图形若不经过任何修饰和标注，那么这样的图形就不会给人以赏心悦目的感觉，因此，必须对图形加以控制并进行适当的标注。本节将详细讨论在 MATLAB 中如何控制及标注图形。

### 7.2.1 线型、点型及颜色的控制

可以在 MATLAB 提供的绘图命令中通过加入控制字符串控制所绘图形的颜色（红色、绿色…）、线型（实线、虚线…）、数据点的标记形式（*、+、…）以及线型的粗细。以 fplot 命令为例，当需要控制图形的线型、点型及颜色时，只需在 fplot 命令中加上控制字符串'color-linestyle-marker'即可。其中 color、linestyle 和 marker 分别代表曲线的颜色、线型以及数据点的点型。它们的取值情况分别如表 7-1、7-2 和 7-3 所示。

表 7-1 颜色控制字符

| 字 符 | 颜 色 | RGB 值 |
|---|---|---|
| b/blue | 兰色 | 0 0 1 |
| c/cyan | 青色 | 0 1 1 |
| g/green | 绿色 | 0 1 0 |
| k/black | 黑色 | 0 0 0 |
| m/magenta | 洋红 | 1 0 1 |
| r/red | 红色 | 1 0 0 |
| w/white | 白色 | 1 1 1 |
| y/yellow | 黄色 | 1 1 0 |

注意：关于什么是 RGB 请参阅附录 B 中的相关内容。

表 7-2 线型控制字符

| 字 符 | 线 型 |
|---|---|
| - | 实线 |
| : | 点线 |
| -. | 点划线 |
| -- | 虚线 |
| 无 | 不绘制图线 |

表 7-3 点型控制字符

| 字 符 | 点 型 | 字 符 | 点 型 |
|---|---|---|---|
| . | 点 | v | 顶点指向下方的三角形 |
| o | 圆圈 | < | 顶点指向左边的三角形 |
| x | 差号 × | > | 顶点指向右方的三角形 |
| + | 十字标号 | ^ | 顶点指向上方的三角形 |
| * | 星号 | p | 五角星 |
| s | 方块 | h | 六角星 |
| D | 钻石形 | 无 | 无点 |

【例 7-15】

在 7.1 节中已经讲过 MATLAB 提供了一个名为 humps 的内置函数，其数学表达式为：humps(x)=$\frac{1}{(x-0.3)^2+0.01}+\frac{1}{(x-0.9)^2+0.04}-6$。试绘制 humps(x)在区间[-1,5]上的图形。要求绘出的图线为钻石形的红色实线。可以执行命令：

```
fplot(@humps,[-1,5],'r-d')
```

执行以上命令所绘制的图形如图 7-36 所示。

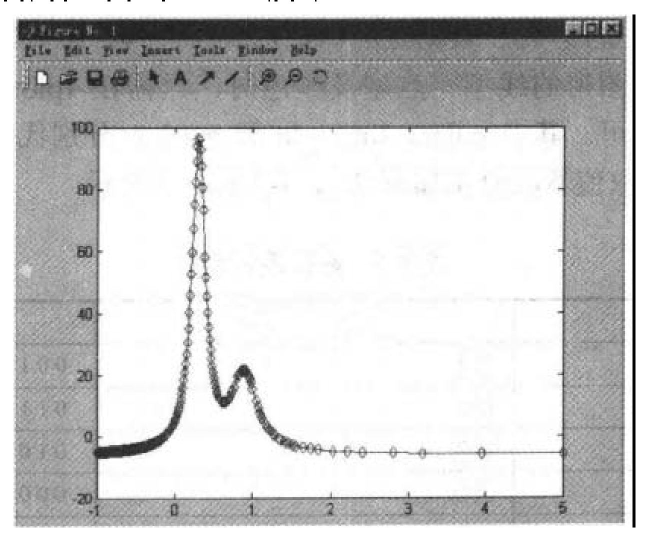


图 7-36 humps(x)在区间[-1,5]上绘制的图形

【例 7-16】

在区间[-2π,2π]上绘制 sin(x)的图形。要求线型为：兰色的点划线。在 MATLAB 命令窗口执行以下命令：

```
fplot(@sin,[-2*pi 2*pi],'b-.')          %绘制 sin 函数在区间[-2π,2π]上的图形
```

以上命令的执行结果如图 7-37 所示。

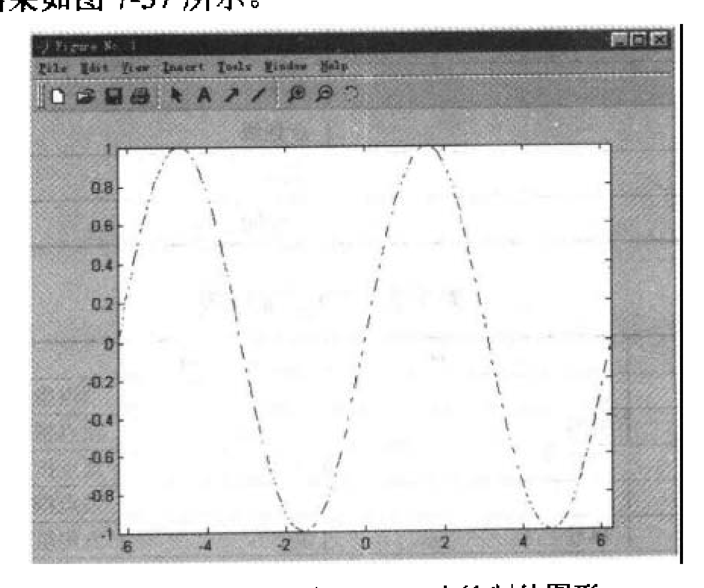


图 7-37 sin(x)在区间[-2π,2π]上绘制的图形

### 7.2.2 线条粗细的控制

以 plot 命令为例，只需在 plot 命令中加入控制字符 LineWidth 就可以控制图线的粗细了。其调用格式为：

- plot(x,y, 'LineWidth',n)：改变 $n$ 的值就可以改变线条的粗细。

【例 7-17】

在区间[0,4]上绘制函数 $y=e^{-x^2}$ 的图像，要求所绘制的图形由区间[0,4]上的 50 个等间距点用红色虚线连接而成，且要求用“*”号标记数据点，线宽为 5 个单位。则可以执行以下程序：

```
x=linspace(0,4,50);          %将区间[0,4]分成 50 个点
y=exp(-x.^2);
plot(x,y,'r--*','LineWidth',5)       %红色虚线，以*标记数据点，线宽为 5
```

以上命令的执行结果如图 7-38 所示。

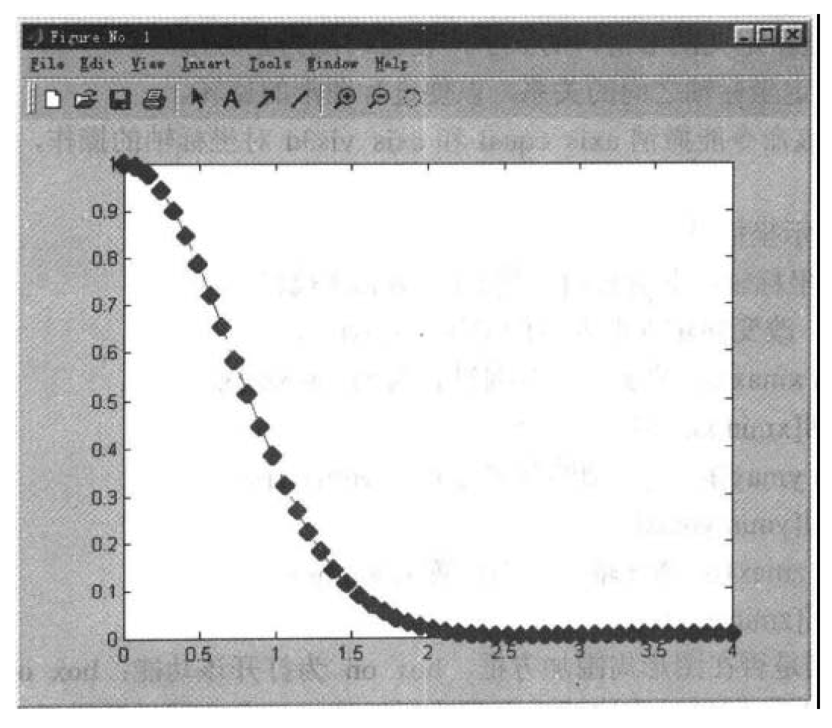



图 7-38 例 7-17 的执行结果

### 7.2.3 坐标轴的控制及窗口缩放

MATLAB 可以根据用户的需要，完美地实现坐标轴的设置，并且它还具有良好的窗口缩放功能。本节将介绍实现上述功能的有关命令。

- axis([xmin xmax ymin ymax])：将图形的 $x$ 轴范围设为从 xmin 到 xmax，把 $y$ 轴范围设为从 ymin 到 ymax。
- axis([xmin xmax ymin ymax zmin zmax])：将图形的 $x$ 轴范围设为从 xmin 到 xmax，把 y 轴范围设为从 ymin 到 ymax，把 $z$ 轴范围设为从 zmin 到 zmax。
- axis('str')或者 axis str：根据控制字符串 str 的值对坐标轴进行控制。其中 str 可取以下

值：

manual：固定坐标轴的刻度。如果当前图形窗口为 hold 打开状态，则后面的图形将采用相同的刻度。

auto：把坐标轴的取值范围重新设置为系统默认状态。

equal：把 $x$ 轴和 $y$ 轴设成同样的刻度增量（注意：实际上是三个坐标轴具有相同的增量）。

tight：将坐标轴设置在数据点范围之内。即此时只绘制包含数据点坐标部分的图形。

fill：该命令用于把 3 个坐标轴的取值范围[xmin xmax ymin ymax zmin zmax]分别设置为绘图所用数据在相应方向的最小、最大值，从而使其能够匹配数据集的范围。

ij：此时把二维图的坐标点设置在图形窗口的左上角。坐标轴 $i$ 垂直向下，坐标轴 $j$ 水平向右。

xy：把二维图的坐标轴恢复为系统的默认状态，即笛卡儿坐标系的设置状态。

image：与 equal 相同。特别是在用 plot 命令创建的坐标轴中，该命令还能起到 axis tight 的作用。

square：重新定义图形窗口的大小，使窗口成为正方形。

vis3d：锁定坐标轴之间的关系，以便进行图形的旋转。

normal：该命令能撤消 axis equal 和 axis vis3d 对坐标轴的操作，使图形窗口恢复到标准大小。

off：不显示坐标轴。

on：显示坐标轴，此时也可以用 axis normal 代替。

- axis(h,…)：改变向量 $h$ 中列出的坐标轴句柄。
- xlim([xmin xmax])：将 $x$ 轴的范围设置为 xmin~xmax。
- xlim：返回[xmin xmax]。
- ylim([ymin ymax])：将 $x$ 轴的范围设置为 ymin~ymax。
- ylim：返回[ymin ymax]。
- zlim([zmin zmax])：将 $z$ 轴的范围设置为 zmin~zmax。
- zlim：返回[zmin zmax]。
- box：控制是否在图形周围加方框。box on 为打开该功能；box off 为关闭该功能。而只使用 box 则可以在这两者之间进行切换（该命令同样适用于三维图形）。
- datetick(axis,format)：根据日期格式 format 来格式化位于坐标轴 axis 上的文本。参数 axis 可以是 $x$、$y$ 或 $z$（缺省时为 $x$）。
- dragrect(x,step)：允许用户在屏幕上拖动矩形，这些矩形是由 n×4 的矩阵 $X$ 中每一行确定的。如果给出 step 则只能在 step 大小的偶数次内拖动矩形。
- grid on：在图形窗口中画出网格线（如果图形是用极坐标绘制的，那么执行该命令后网格线也将按照极坐标形式画出）。
- grid off：从图形窗口中消除网格线。
- grid：用于实现在 grid on 和 grid off 之间进行切换。
- zoom on：该命令使系统处于放大状态，此时可以有两种方法实现图形的放大：用鼠标单击需要放大的部分，可将该处放大一倍。这一操作可以连续进行。当鼠标右击

时，图形将退回到上一次的放大状态；用鼠标拖出一个方框，系统将放大被方框框住的部分。

- zoom off：关闭 zoom 功能，用于使系统转回非放大状态，但前面放大所得的结果不会改变。
- zoom out：将系统返回到非放大状态并将图形恢复至原状。
- zoom reset：使系统记住此时图形的放大状态，当使用 zoom out 时，图形将不返回到原状，而是返回到 reset 时的放大状态。
- zoom：该命令可以用于在 zoom in 和 zoom off 之间进行切换。
- zoom xon：该命令只对 $x$ 轴起放大作用。
- zoom yon：该命令只对 $y$ 轴起放大作用。
- zoom(factor)：用于按照参数 factor 的值对图形进行缩放。当 factor>1 时，系统将把图形放大 factor 倍。当 factor≤1 时，系统将图形缩小 factor 倍。
- zoom(fig,option)：在选择非当前图形窗口 fig 以后，将该窗口设为放大状态。其中 option 可以取 on、off 以及 xon 等参数。

以上介绍的命令数量很多，功能各异。一一给出它们的使用例子是不可能的，读者只需了解各个命令的相应功能，在需要的时候就可以尝试使用。在这里只通过一个例子，说明 axis 命令和 grid 命令的使用方法。

【例 7-18】

先用下面的命令绘制一个单位圆：

```
t=0:0.2:2*pi;
x=sin(t);
y=cos(t);
plot(x,y,'-')
```

以上命令的执行结果如图 7-39 所示。

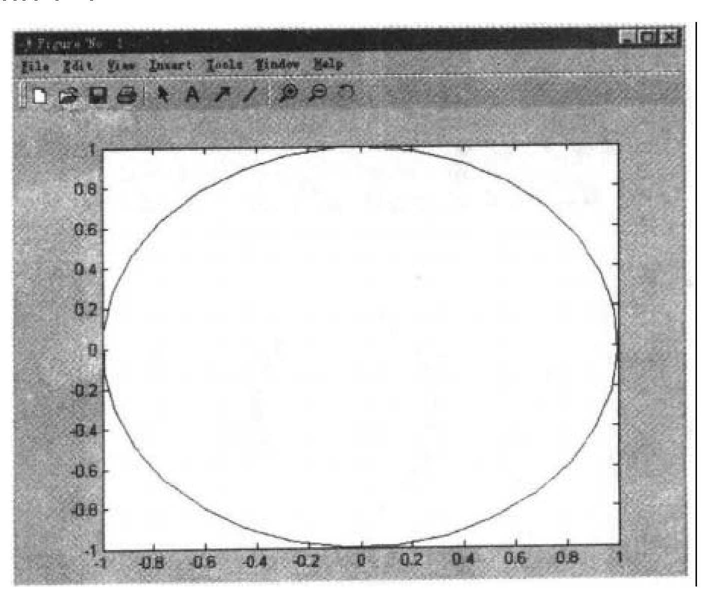



图 7-39　不加修饰的单位圆

从图 7-39 中可以看出，所绘制的单位圆更像一个椭圆。现在再执行以下命令就会使图

形显示的效果得到大大改观。

```
axis square
grid on
```

执行以上命令后图形如图 7-40 所示。

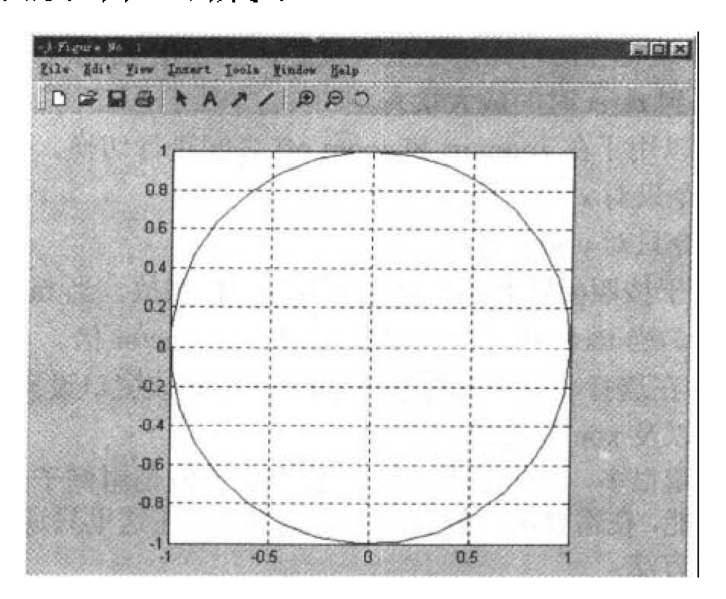



图 7-40 坐标轴修饰经过后的单位圆

可见此时的单位圆已经比较令人满意了。

为了理解坐标轴的设定对显示图形的影响，可以执行以下命令：

```
axis normal        %此时图形又回到图 7-39 的效果，但还有网格线存在
grid off           %这时图形完全回到图 7-39 所示的效果
axis([-3 3 -2 2])  %此时图形变成椭圆了
```

执行以上命令后得到的结果如图 7-41 所示。

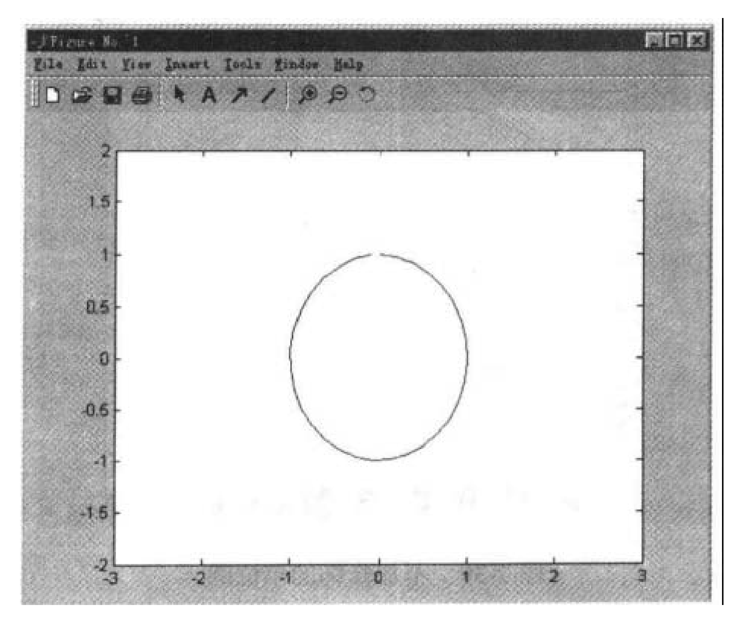



图 7-41 坐标轴重新设定后的单位圆

当把坐标轴分别设置为 axis equal 和 axis tight 时，所绘制的图形如图 7-42 和图 7-43 所示。

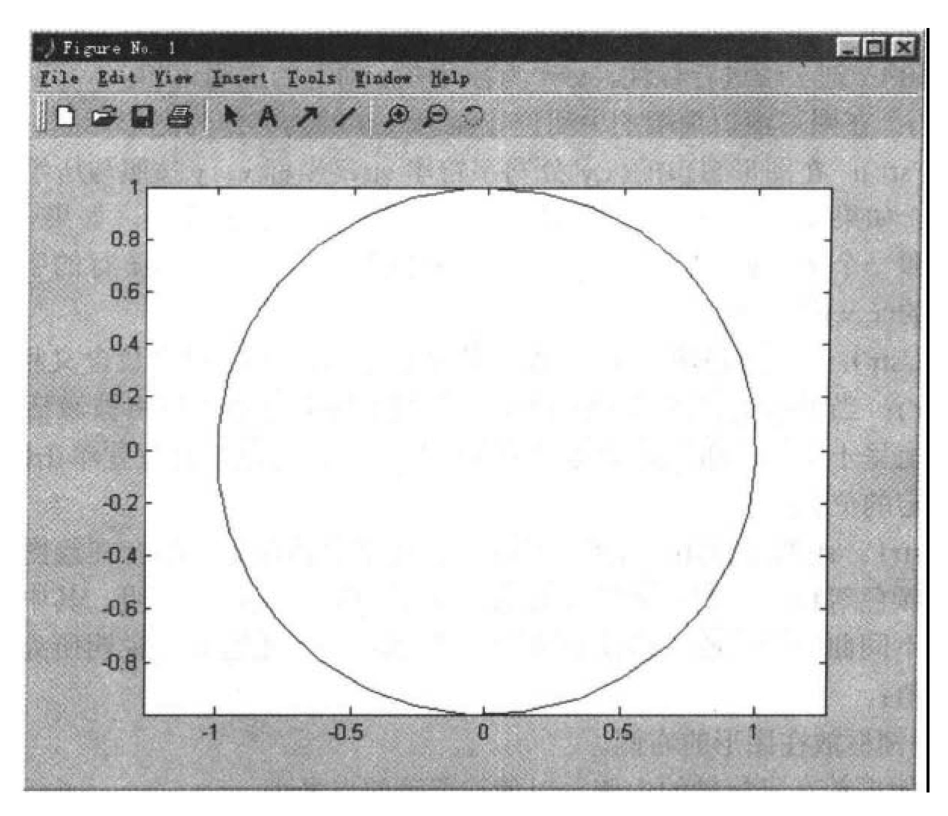



图 7-42　由 axis equal 得到的单位圆

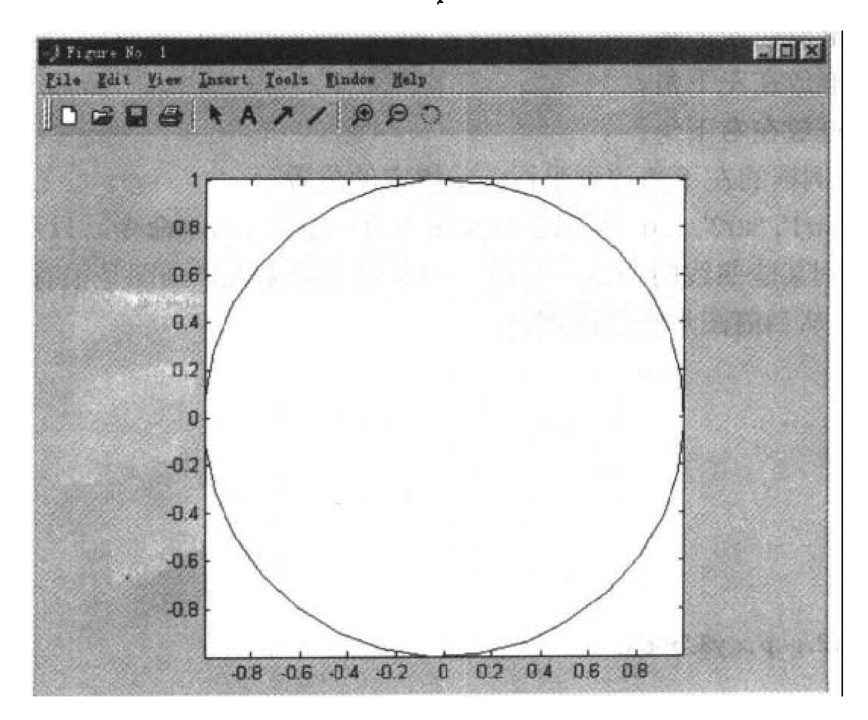



图 7-43　由 axis tight 得到的单位圆

### 7.2.4　图形标注

利用前面介绍的命令虽然可以完成一幅图形的绘制，但是这样的图形还算不上完美。因

为前面的图形都没进行标注。接下来，将介绍有关图形标注方面的知识。

- xlabel('text')：对 $x$ 轴进行标注，text 为标注在 $x$ 轴上的说明语句。
- ylabel('text')：对 $y$ 轴进行标注，text 为标注在 $y$ 轴上的说明语句。
- zlabel('text')：对 $z$ 轴进行标注，text 为标注在 $z$ 轴上的说明语句。
- title('text')：在图形窗口顶端的中间位置输出 text 的内容作为图形标题。
- text(x,y, 'str')：在图形窗口的(x,y)处写字符串 str，坐标 $x$、$y$ 按照与所绘图形相同的刻度给出。如果 $x$、$y$ 为向量，那么字符串将写在$(x_i,y_i)$的位置上。如果 str 是一个字符串向量即一个字符矩阵，并且与 $x$、$y$ 有相同的行数，那么第 $i$ 行的字符串将写在图形窗口的$(x_i,y_i)$位置上。
- text(x,y,z,'str')：三维图的标注，参数含义和 text(x,y, 'str')中的参数含义相同。
- gtext('str')：当程序运行到该条命令时，系统的当前图形窗口将被激活，此时鼠标在该窗口上呈十字形，用来提示标注位置。用户可以在期望的位置单击鼠标，str 就被写到指定的位置。
- legend('str1', 'str2',...ops)：当在一幅图中出现多条曲线时，结合在绘图时所用的不同线型和颜色等特点，用户可以用该命令对不同的图例进行说明。其中 str1,str2,...是对图中不同曲线的说明。该命令中的参数 ops 用于指定图例说明所处的位置，可取以下数值：

  -1：将图例框放在图形的右侧。

  0：将图例框放在坐标轴的内侧，以使被覆盖的点最少。

  1：将图例框放在右上角（系统默认情况）。

  2：将图例框放在左上角。

  3：将图例框放在左下角。

  4：将图例框放在右下角。
- [x,y]：将图例框的左下角放到坐标(x,y)指定的位置。
- legend(H, ('str1', 'str2',...)：类似于 legend('str1', 'str2',...ops)命令。只不过该命令返回句柄 H 以得到某些曲线的合适字符串。这对用矩阵输入来画图是很有必要的。
- legend off：从当前图形中清除图例。

【例 7-19】

执行以下程序：

```
x=0:0.2:5;
y1=sin(x);
y2=x/2;
y3=cos(2*x+1);
plot(x,y1,'r-+',x,y2,'g-.p',x,y3,'b--o');
title('3 个函数')
xlabel('横坐标轴 X')
ylabel('纵坐标轴 Y')
```

执行以上命令后可以得到如图 7-44 所示的图形。

为了进行图例标注，再执行以下命令：

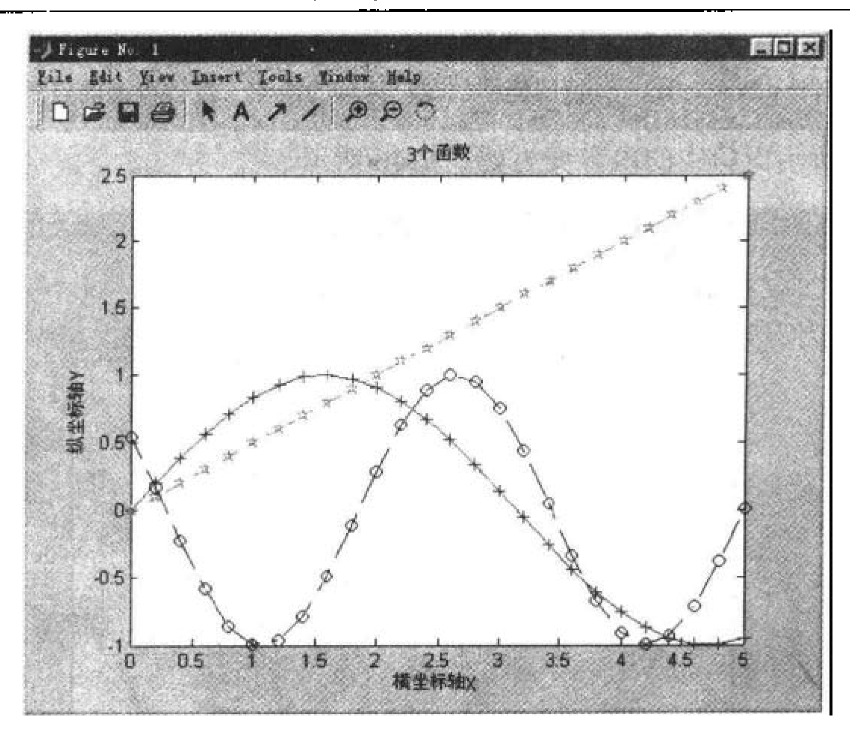



图 7-44 进行了图形标注的图形

```
axis([-0.5 5.5 -1.5 2.5]);               %改变坐标轴的范围
legend('sinx','x/2','cos(2x+1)')          %进行图例标注
gtext('Wonderful!')                       %用鼠标输入字符串 Wonderful!
```

以上命令的执行结果如图 7-45 所示。

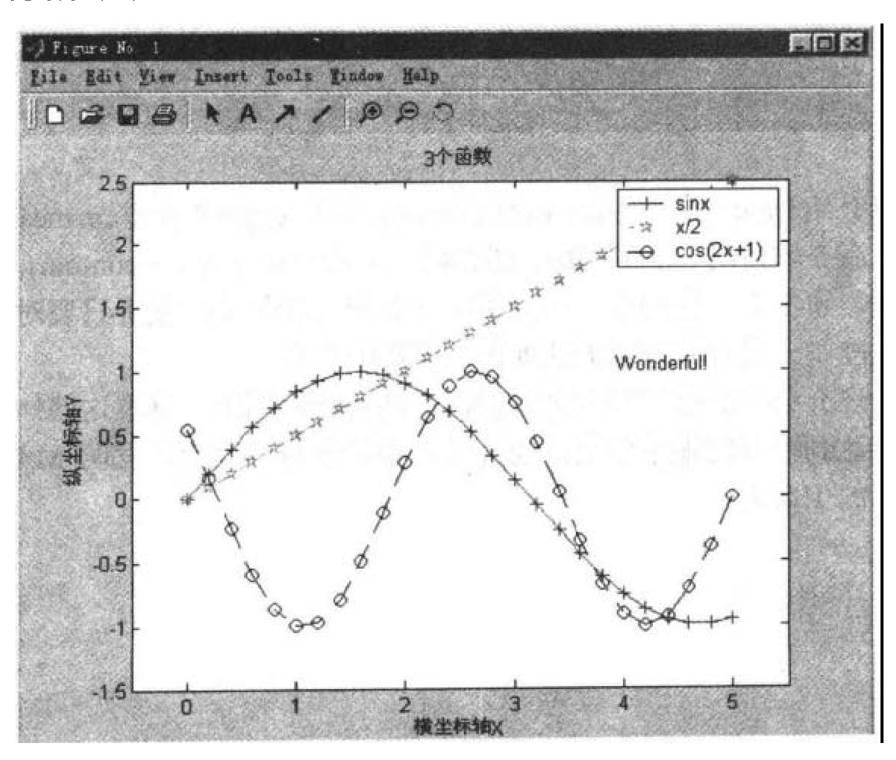



图 7-45 进行了图例标注的图形

从图 7-45 可以看出，图例框默认情况下是放在右上角的，现在重新设置一下，使它不挡住所绘制的图形。可以执行以下命令：

```
legend off                              %取消图例框
legend('sinx','x/2','cos(2x+1)',0)      %将图例框放在坐标轴的内侧，使被覆盖的点最少
```

此时图 7-45 所示窗口中的内容变为如图 7-46 所示。

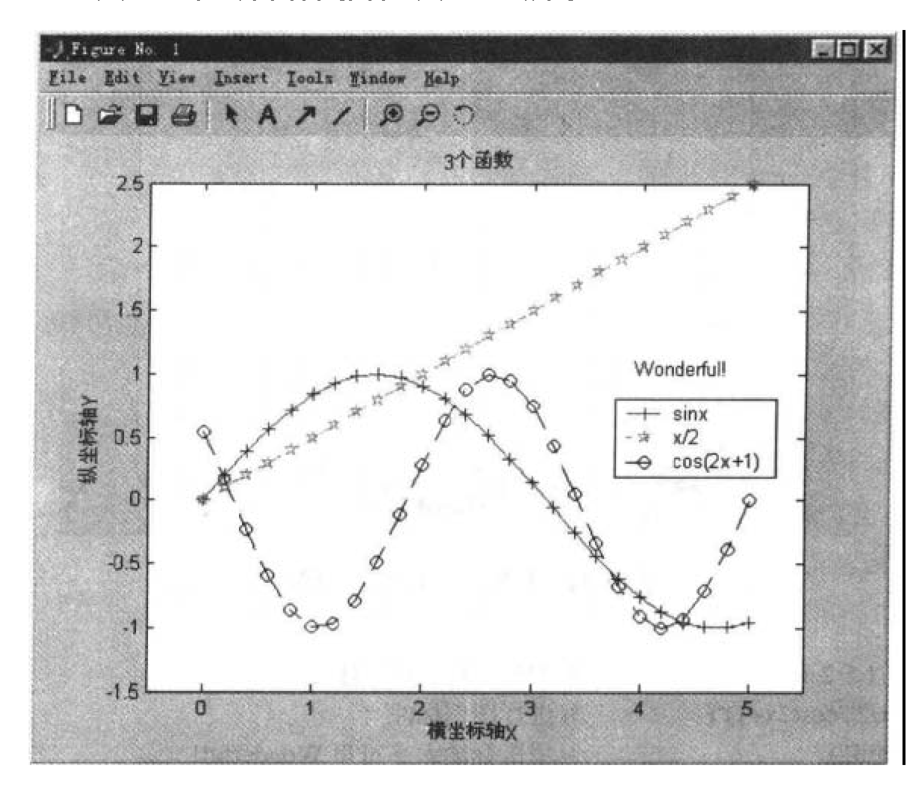



图 7-46 重新进行图例标注的图形

## 7.3 小结

本章在重点介绍 fplot 命令、ezplot 命令、ezpolar 命令、ezplot3 命令、ezmesh 命令、ezcontour 命令以及 ezsurf 命令的使用方法的同时，还介绍了 ezmeshc 命令、ezcontourf 命令以及 ezsurfc 命令的使用方法。事实上，这些命令的调用方法是极其相似的，读者只要对其中一种命令做到熟练掌握，就能举一反三熟练掌握其他图形的绘制命令。

另外，本章还介绍了如何对图形进行控制和如何标注图形，掌握这些知识会使所绘制的图形更为美观，达到较好的视觉效果。通过对本章的学习，读者会发现 MATLAB 在图形处理方面具有强大的生命力。

# 附　　录

## 附录 A　怎样获取符号运算函数及其命令的帮助信息

在 MATLAB 中可以通过两种方法获得符号运算函数及其命令的有关帮助信息。

### 1．通过查看 MATLAB 的附带函数说明文档来获得帮助信息

这种帮助方式可以通过使用网络浏览器（如 IE、Netscape）查阅 HTML 文档的方法实现。也可以利用 Acrobat Reader 阅读浏览器查看 PDF 文档获得帮助信息。其中，HTML 文档的主页 helpdesk.html 保存在 MATLAB 所在目录下的 help 子目录内；PDF 文档则保存在 help 的子目录 pdf_doc 内。

### 2．使用 MATLAB 的帮助命令 help

help 的调用格式是：

● help function

其中 function 为待查看的 MATLAB 函数名。

用户也许会发现用上述两种方法查看符号运算函数的帮助信息时，往往得到的都是数值量运算的有关帮助信息，这是因为：MATLAB 中数值量和符号量的很多运算函数名都是相同的。因此当调用符号运算命令时，MATLAB 必须对其中某些命令重新加载（即把它们由适于数值量的计算转化为适用于符号量的计算）。但是在命令 help function 中，function 函数并没被重载，因而通过 help function 获得的帮助信息也就只能是针对数值量进行计算的函数了。例如，想了解求导数命令 diff 在符号运算中的使用情况，可以在 MATLAB 工作空间中执行以下命令：

```
help diff
```

得到的执行结果为：

```
DIFF Difference and approximate derivative.
   DIFF(X), for a vector X, is [X(2)-X(1)  X(3)-X(2) ... X(n)-X(n-1)].
   DIFF(X), for a matrix X, is the matrix of row differences,
      [X(2:n,☺ - X(1:n-1,☺].
   DIFF(X), for an N-D array X, is the difference along the first
      non-singleton dimension of X.
   DIFF(X,N) is the N-th order difference along the first non-singleton
      dimension (denote it by DIM). If N >= size(X,DIM), DIFF takes
      successive differences along the next non-singleton dimension.
   DIFF(X,N,DIM) is the Nth difference function along dimension DIM.
      If N >= size(X,DIM), DIFF returns an empty array.
   Examples:
```

```
        h = .001; x = 0:h:pi;
        diff(sin(x.^2))/h is an approximation to 2*cos(x.^2).*x
        diff((1:10).^2) is 3:2:19
        If X = [3 7 5
                0 9 2]
        then diff(X,1,1) is [-3 2 -3], diff(X,1,2) is [4 -2
                                                         9 -7],
        diff(X,2,2) is the 2nd order difference along the dimension 2, and
        diff(X,3,2) is the empty matrix.
    See also GRADIENT, SUM, PROD.
 Overloaded methods
    help sym/diff.m
    help char/diff.m
    help fints/diff.m
```

可见这样获得的帮助信息是针对数值量的运算的。但应当注意的是，帮助信息的最后 3 行所显示的内容，它们为用户提供了重载 diff 命令的一种方法。同时还可得知，diff 命令重载后可以应用于符号量（sym）和字符量（char）。

完成 diff 命令关于符号量运算的有关帮助信息可以通过以下命令之一来实现：

```
help sym/diff.m
```

或者

```
help sym/diff
```

其显示结果为：

```
DIFF    Differentiate.
    DIFF(S) differentiates a symbolic expression S with respect to its
    free variable as determined by FINDSYM.
    DIFF(S,'v') or DIFF(S,sym('v')) differentiates S with respect to v.
    DIFF(S,n), for a positive integer n, differentiates S n times.
    DIFF(S,'v',n) and DIFF(S,n,'v') are also acceptable.
    Examples;
        x = sym('x');
        t = sym('t');
        diff(sin(x^2)) is 2*cos(x^2)*x
        diff(t^6,6) is 720.
    See also INT, JACOBIAN, FINDSYM.
```

应当注意的是，以上介绍的两种获得帮助信息的方法均局限于 MATLAB 的内部命令和函数，这也就限制了它们的使用范围。当所查询的函数是 MAPLE V“引擎”中的函数时，help 命令是无法查找出相关的帮助信息。这时可以借助于 mhelp 命令来获得有关帮助信息，其使用格式和 help 命令的使用格式完全相同。需要注意的是，mhelp 命令获得的是 MAPLE V 中函数命令的有关帮助信息。

例如，要查看 bernulli 命令的帮助信息，可以在 MATLAB 工作空间执行以下命令：

mhelp bernoulli

则得到的执行结果为：

```
bernoulli or numtheory[B] - Bernoulli numbers and polynomials
Calling Sequence:
      bernoulli(n)
      bernoulli(n, x)
      numtheory[B](n)
      numtheory[B](n, x)
Parameters:
      n - a non-negative integer
      x - an expression
Description:
- The bernoulli function computes the nth Bernoulli number, or the nth Bernoulli
  polynomial, in the expression x. The nth Bernoulli polynomial B(n,x) is
  defined by the exponential generating function:
      t*exp(x*t)/(exp(t)-1) = sum( B(n,x)/n!*t^n, n=0..infinity ) .
- The nth Bernoulli number B(n) is defined as B(n,0).
Examples:
> bernoulli(4);
                                   -1
                                   --
                                   30
> bernoulli(4,x);
                               2     4     3
                        -1/30 + x  + x  - 2 x
> bernoulli(4,1/2);
                                 7/240
> with(numtheory):
Warning, new definition for order
> B(6);
                                  1/42
> B(6,x);
                             2          4     6     5
                 1/42 - 1/2 x  + 5/2 x  + x  - 3 x
> B(6,1/2);
                                  -31
                                  ----
                                  1344
See Also:
euler, inifcns
```

读者可以通过在 MATLAB 工作空间中键入 mhelp diff 后按 Enter 键的办法，来体验使用 mhelp 命令查看 MATLAB 内部函数的帮助信息的调用格式。

# 附录 B 什么是 RGB

MATLAB 使用色图来绘制表面图形。色图是一个 $m\times3$ 的矩阵，该矩阵的每一行代表一种颜色，而第一列给出红颜色（Red）的数值，第二列给出绿颜色（Green）的数值，第三列则给出蓝颜色（Blue）的数值。即色图矩阵的每一行代表一个 RGB 值，这样一个 $m\times3$ 的色图矩阵就可以定义 $m$ 种颜色。由物理学的知识可知，其他的颜色值都可以由 RGB 值的多少来确定。最为常见的颜色配比方案除了表 7-1 给出的以外，表 B-1 所列的几种 RGB 配比方案也是较为常见的。

表 B-1 较为常用的颜色配比方案

| R 值 | G 值 | B 值 | 颜色 |
|---|---|---|---|
| 0.5 | 0.5 | 0.5 | 灰色（Gray） |
| 0.5 | 0 | 0 | 暗红色（Dark Red） |
| 1 | 0.62 | 0.4 | 红铜色（Copper） |
| 0.49 | 1 | 0.83 | 碧绿色（Aquamayine） |

注：其他的颜色配比方案参见表 7-1。

知道了上述 RGB 的值与颜色的对应情况，就可以通过设置不同的 RGB 值，来控制绘图时所需的颜色了。

# 附录C　MATLAB基本命令和函数目录

MATLAB 6.x版本提供了24类基本命令函数，它们一部分是MATLAB的内部命令，一部分是以m文件形式出现的函数，这些m文件按照类归集于一个子目录中，每个目录除了以m文件表示的函数命令以外，还有一个特殊的contents.m文件，它包含了该目录中各个m文件的简介。每个函数文件中都包含了这一函数的用法指南，因此可以用命令“help fn”来显示有关函数fn的帮助信息（其中fn为m文件），也可以用命令“help dn”来显示该目录下各个函数文件的简要说明（其中dn为目录名）。

附表C-1为基本命令函数的子目录及其含义，表C-2~表C-25列出了MATLAB 6.x版本提供的24类函数的简要说明，以供用户参考。

提示：如果MATLAB 6.1安装在D盘，则该版本提供的24类函数位于D:/matlab6p1/toolbox/matlab目录下；如果MATLAB 6.0安装在D盘，则该版本提供的24类函数位于D:/matlab6R12/toolbox/matlab目录下。

表C-1　MATLAB 6.x版本中提供的24类基本命令函数目录

| 目　录　名 | 命令函数 | 索　　引 |
|---|---|---|
| audio | 音频处理函数 | 表C-2 |
| datafun | 数据分析和傅立叶变换 | 表C-3 |
| datatypes | 数据类型和结构 | 表C-4 |
| demos | 演示示例 | 表C-5 |
| elfun | 基本数学函数 | 表C-6 |
| elmat | 基本矩阵和矩阵操作 | 表C-7 |
| funfun | 泛函和常微分方程求解器 | 表C-8 |
| general | 通用命令 | 表C-9 |
| graph2d | 2维图形函数 | 表C-10 |
| graph3d | 3维图形函数 | 表C-11 |
| graphics | 图形句柄函数 | 表C-12 |
| iofun | 文件输入/输出函数 | 表C-13 |
| lang | 语言结构 | 表C-14 |
| matfun | 矩阵函数以及数值线性代数 | 表C-15 |
| ops | 操作符和特殊字符 | 表C-16 |
| polyfun | 插值和多项式 | 表C-17 |
| sparfun | 稀疏矩阵函数 | 表C-18 |
| specfun | 特殊数学函数 | 表C-19 |
| specgraph | 特殊图形函数 | 表C-20 |
| strfun | 字符串函数 | 表C-21 |

（续）

| 目 录 名 | 命 令 函 数 | 索 引 |
| --- | --- | --- |
| timefun | 时间和日期函数 | 表 C-22 |
| uitools | 图形用户界面（GUI）工具 | 表 C-23 |
| verctrl | 版本控制函数 | 表 C-24 |
| Winfun | Windows 操作系统界面文件（DDE/ActiveX） | 表 C-25 |

### 表 C-2 MATLAB 中的音频处理函数

| 功 能 类 别 | 命 令 函 数 | 说 明 |
| --- | --- | --- |
| 音频输入/输出对象 | 音频输入/输出对象 | |
| | audioplayer | Windows 音频播放对象 |
| | audiorecorder | Windows 音频记录对象 |
| 音频硬件驱动器 | sound | 变矢量为音频信号 |
| | soundsc | 量化矢量并演奏为音频 |
| | wavplay | 用 Windows 音频输出设备播放声音 |
| | wavrecord | 用 Windows 音频输入设备记录声音 |
| 音频文件的输入/输出 | auread | 读 NeXT/SUN (".au") 声音文件 |
| | auwrite | 写 NeXT/SUN (".au")声音文件 |
| | wavread | 读 Microsoft WAVE (".wav")声音文件 |
| | wavwrite | 写 Microsoft WAVE (".wav")声音文件 |
| 一般音频函数 | lin2mu | 变线性音频信号为 mu-law 编码的音频信号 |
| | mu2lin | 变 mu-law 编码的音频信号为线性音频信号 |
| 音频数据举例（为 MAT 文件） | chirp | 鸟叫声（1.6s，8192 Hz） |
| | gong | 打锣声（5.1s，8192 Hz） |
| | handel | 合唱赞美诗的声音（8.9s，8192 Hz） |
| | laughter | 人群发出的笑声（6.4s，8192 Hz） |
| | splat | 噼啪声之后紧随的鸟叫声 |
| | train | 火车鸣笛声（1.5s，8192Hz） |

### 表 C-3 MATLAB 中的数据分析和傅立叶变换

| 功 能 类 别 | 命 令 函 数 | 说 明 |
| --- | --- | --- |
| 基本操作 | cumprod | 求元素的累积乘积 |
| | cumsum | 求元素的累积和 |
| | cumtrapz | 梯形数值积分累积值 |
| | hist | 柱状图 |
| | histc | 柱状图计算 |
| | max | 取最大分量 |
| | mean | 求均值 |
| | median | 求中值 |
| | min | 取最小分量 |
| | prod | 求元素的积 |

（续）

| 功能类别 | 命令函数 | 说　明 |
| --- | --- | --- |
| 基本操作 | sort | 按照升序排列 |
| | sortrows | 对行进行升序排列 |
| | std | 求标准差 |
| | sum | 求元素的和 |
| | trapz | 利用梯形法计算数值积分 |
| | var | 求方差 |
| 有限插分 | del2 | 离散 Laplace（拉普拉斯）算子 |
| | diff | 计算插分和近似微分 |
| | gradient | 计算近似梯度 |
| 滤波和卷积 | conv | 卷积和多项式乘法 |
| | conv2 | 二维卷积 |
| | convn | N 维卷积 |
| | deconv | 反卷积和多项式除法 |
| | detrend | 消除线性趋势 |
| | filter | 一维数字滤波器 |
| | filter2 | 二维数字滤波器 |
| 傅立叶变换 | fft | 离散傅立叶变换 |
| | fft2 | 二维离散傅立叶变换 |
| | fftn | N 维离散傅立叶变换 |
| | fftshift | 将零点平移至频谱中心 |
| | ifft | 离散逆傅立叶变换 |
| | ifft2 | 二维离散傅立叶变换 |
| | ifftn | N 维离散逆傅立叶变换 |
| | ifftshift | 为 fftshift 的逆变换，也就是将零点平移至时间中心 |

**表 C-4　MATLAB 中的数据类型和结构**

| 功能类别 | 命令函数 | 说　明 |
| --- | --- | --- |
| 数据类型（类） | Cell | 创建单元数组 |
| | char | 创建字符型数组（字符串） |
| | double | 转换为双精度 |
| | function_handle | 函数句柄数组 |
| | javaArray | 构建 Java 数组 |
| | javaMethod | 调用 Java 方法 |
| | javaObjec | 调用 Java 对象构造器 |
| | inline | 建立内联对象 |
| | int8 | 转换为有符号的 8 位整数 |
| | int16 | 转换为有符号的 16 位整数 |
| | int32 | 转换为有符号的 32 位整数 |

（续）

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 数据类型（类） | single | 转换为单精度 |
| | sparse | 创建稀疏矩阵 |
| | struct | 创建或者转换为结构数组 |
| | uint8 | 转换为无符号 8 位整数 |
| | uint16 | 转换为无符号 16 位整数 |
| | uint32 | 转换为无符号 32 位整数 |
| 多维数组函数 | cat | 连接数组 |
| | ipermute | 逆序排列数组的维数 |
| | ndgrid | 为 N 维函数或插值产生数组 |
| | ndims | 维数 |
| | permute | 排列数组的维数 |
| | shiftdim | 移维 |
| | squeeze | 移去单一的维数 |
| 单元数组函数 | cell | 创建单元数组 |
| | celldisp | 显示单元数组的内容 |
| | cellfun | 对单元数组每一元素取函数 |
| | cellplot | 图形方式显示单元数组 |
| | cell2struct | 将单元数组转换为结构数组 |
| | deal | 将输入分配给输出 |
| | iscell | 判断数组是否为单元数组 |
| | num2cell | 将数值数组转换为单元数组 |
| | struct2cell | 将结构数组转换为单元数组 |
| 结构函数 | fieldnames | 获取结构的域名 |
| | getfield | 获取结构的域 |
| | isfield | 判断域是否为结构数组 |
| | isstruct | 判断是否为结构 |
| | rmfield | 删除结构的域 |
| | setfield | 设置结构的域 |
| | struct | 创建或者转换为结构数组 |
| 函数句柄的有关函数 | @ | 创建函数句柄 function_handle |
| | functions | 列出与函数句柄相关联的表 |
| | func2str | 将函数句柄数组转换成字符串 |
| | str2func | 将字符串转换成函数句柄数组 |
| 面向对象程序设计函数 | Class | 创建对象或者返回对象的类 |
| | Inferiorto | 子类 |
| | Isa | 判断对象是否是一个给定的类 |
| | Isjava | 判断是否为 Java 对象 |
| | isobject | 判断是否为 MATLAB 对象 |
| | methods | 列出类方法的名称和属性 |

（续）

| 功能类别 | 命令函数 | 说　明 |
|---|---|---|
| 面向对象程序设计函数 | methodsview | 观察类方法的名称和属性 |
| | struct | 将对象转换为结构数组 |
| | substruct | 为 SUBSREF/SUBSASGN 创建结构参数 |
| | superiorto | 高级类 |
| 可以重载的操作符 | and | 可以重载的 a&b 方法 |
| | colon | 可以重载的 a:b 方法 |
| | ctranspose | 可以重载的 a'方法 |
| | end | 可以重载的 a(end)方法 |
| | eq | 可以重载的 a==b 方法 |
| | ge | 可以重载的 a>=b 方法 |
| | gt | 可以重载的 a>b 方法 |
| | Horzcat | 可以重载的水平连接方法，如[a b] |
| | ldivide | 可以重载的数组左除，如 a.\b |
| | le | 可以重载的 a<=b 方法 |
| | loadobj | 当从.mat 文件载入对象时调用该函数 |
| | lt | 可以重载的 a<b 方法 |
| | minus | 可以重载的减法，如 a-b |
| | mldivide | 可以重载的矩阵左除，如 a\b |
| | mpower | 可以重载的矩阵求幂，如 a^b |
| | mrdivide | 可以重载的矩阵右除，如 a/b |
| | mtimes | 可以重载的矩阵乘法，如 a*b |
| | ne | 可以重载的 a~=b 方法 |
| | not | 可以重载的~a 方法 |
| | or | 可以重载的 a\|b 方法 |
| | plus | 可以重载的加法，如 a+b |
| | power | 可以重载的数组求幂，如 a.^b |
| | rdivide | 可以重载的数组右除，如 a./b |
| | saveobj | 当把对象存成.mat 文件时调用该函数 |
| | subsasgn | 可以重载的 a(I)=b，a{I}=b 以及 a.field=b 方法 |
| | subsindex | 可以重载的 x(a)方法 |
| | subsref | 可以重载的 a(I)，a{I}以及 a.field 方法 |
| | times | 可以重载的数组乘法，如 a.*b |
| | transpose | 可以重载的 a.'方法 |
| | uminus | 可以重载的一元减法，如-a |
| | uplus | 可以重载的一元加法，如+a |
| | vertcat | 可以重载的竖直连接方法，如[a;b] |

表 C-5 MATLAB 中的演示示例

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| MATLAB/简介 | demo | 浏览 MATLAB、Toolboxes 和 SIMULINK 的演示示例 |
| MATLAB/矩阵 | airfoil | 显示代表 NASA（美国航空和宇宙航行局）机翼的稀疏矩阵 |
| | buckydem | 演示 Buckminster Fuller 网格球顶连通图 |
| | delsqdemo | 在不同的域上求有限插分拉普拉斯算子 |
| | eigmovie | 对称的特征值动画 |
| | eigshow | 图形演示矩阵的特征值 |
| | intro | 简要介绍 MATLAB 中的基本矩阵操作 |
| | inverter | 演示矩阵求逆 |
| | matmanip | 矩阵操作简介 |
| | rrefmovie | 计算缩减行阶梯阵 |
| | sepdemo | 有限元网格的划分 |
| | sparsity | 演示稀疏排序的效果 |
| | svdshow | 图形演示矩阵的奇异值 |
| MATLAB/数值 | bench | MATLAB Benchmark.（MATLAB 基准） |
| | census | 尽可能预测美国 2000 年的人口 |
| | expm1 | 用 Pade 近似法计算矩阵指数 |
| | expm2 | 用 Taylor 级数法计算矩阵指数 |
| | expm3 | 根据特征值和特征矢量计算矩阵指数 |
| | e2pi | 在 2 维里观察答案：e^pi 和 pi^e 谁大谁小？ |
| | fftdemo | 演示快速傅立叶变换 |
| | fitdemo | 用单形算法实现非线性曲线拟合 |
| | fplotdemo | 绘制函数的图形 |
| | funfuns | 演示函数作用于其他函数的情况 |
| | lotkademo | 演示一个求解常微分方程的例子 |
| | qhulldemo | 离散数据的布置和插值 |
| | Quaddemo | 自适应求积分 |
| | quake | 演示 Loma Prieta 地震 |
| | spline2d | 2 维内演示 GINPUT 和 SPLINE |
| | sunspots | 对 FFT 来说，答案为 11.08。问题是什么呢？ |
| | zerodemo | 用 fzero 函数寻找零点 |
| MATLAB/可视化 | colormenu | 选择颜色板 |
| | cplxdemo | 复变数的函数映射 |
| | earthmap | 观察地球的地形 |
| | grafcplx | 演示在 MATLAB 绘制复数函数图形 |
| | graf2d | 2 维绘图：演示在 MATLAB 绘制 XY 图形 |
| | graf2d2 | 3 维绘图：演示在 MATLAB 绘制 XYZ 图形 |
| | imagedemo | 演示 MATLAB 的图像功能 |

（续）

| 功能类别 | 命令函数 | 说　明 |
|---|---|---|
| MATLAB/可视化 | imageext | 图像的颜色板：改变和旋转颜色板 |
| | lorenz | 绘制在 Lorenz 混沌吸引子周围的轨道 |
| | penny | 演示币面 |
| | xfourier | 用图形演示傅立叶级数展开 |
| | xpklein | 演示 Klein（克莱因）瓶 |
| | xpsound | 观察声音：演示 MATLAB 的声音功能 |
| | vibes | 演示振动动画：L 形膜的振动 |
| MATLAB/语言 | graf3d | 演示表面图的图形处理功能 |
| | hndlaxis | 演示坐标轴的图形处理功能 |
| | hndlgraf | 演示图线的图形处理功能 |
| | xplang | 介绍 MATLAB 语言 |
| MATLAB/微分方程 | odedemo | 演示 MATLAB 微分方程求解器 |
| | odeexamples | 浏览 MATLAB 的 ODE/DAE/BVP/PDE 例子 |
| MATLAB/ODEs（常微分方程） | ballode | 演示跳跃的球 |
| | brussode | 一个化学反应（Brusselator）的刚性问题建模 |
| | burgersode | 用移动网格技术求解 Burger 方程 |
| | fem1ode | 演示依赖于时间的质量矩阵刚性问题 |
| | fem2ode | 演示常质量矩阵的刚性问题 |
| | hb1ode | Hindmarsh 和 Byrn 第一刚性问题 |
| | orbitode | 受约束的 3 体问题 |
| | rigidode | 不受外力作用的刚体欧拉（Euler）方程 |
| | vdpode | 参数化的 van der Pol 方程（当 μ 很大时为刚性） |
| MATLAB/DAEs（微分-代数方程） | amp1dae | 来自于电路的刚性 DAE 方程 |
| | hb1dae | 来自于守恒定律的刚性 DAE 方程 |
| MATLAB/BVPs（边值问题） | mat4bvp | 寻找 Mathieu 方程的第四个特征值 |
| | shockbvp | 解在 x = 0 产生振荡 |
| | twobvp | 恰好有两个解的 BVP 问题 |
| MATLAB/PDEs（偏微分方程） | pdex1 | PDEPE 之例 1 |
| | pdex2 | PDEPE 之例 2 |
| | pdex3 | PDEPE 之例 3 |
| | pdex4 | PDEPE 之例 4 |
| | pdex5 | PDEPE 之例 5 |
| 附加/图库 | cruller | 构造油煎饼状的图形 |
| | klein1 | 构造克莱因瓶 |
| | knot | 管道缠绕成 3 维的结 |
| | logo | 显示 MATLAB 的 L 标识 |
| | modes | 绘制 12 种 L 形的膜 |
| | quivdemo | 演示箭头图函数 |

（续）

| 功能类别 | 命令函数 | 说明 |
|---|---|---|
| 附加/图库 | spharm2 | 构造球面谐波 |
| | tori4 | 箍：构造 4 个连接的圆环 |
| 附加/游戏 | fifteen | 字谜幻灯 |
| | life | Conway 生命游戏 |
| | soma | Soma 魔方游戏 |
| | xpbombs | 扫雷游戏 |
| 附加/其他 | chaingui | 矩阵链乘法优化 |
| | codec | 字母表转换编码器/解码器 |
| | crulspin | 旋转的油煎饼状图形动画 |
| | logospin | 动画演示 MathWorks 旋转标识 |
| | makevase | 产生并绘制一个旋转面 |
| | quatdemo | 旋转四元数 |
| | spinner | 穿透表面的彩线 |
| | travel | “货郎担”问题 |
| | truss | 动画演示弯曲的桥桁架 |
| | wrldtrv | 围绕地球的巨大圆形飞行路径 |
| | xphide | 在运动中观察对象 |
| | xpquad | 演示超二次曲面的绘制 |
| 一般演示/帮助函数 | cmdlnbgn | 设置命令行演示 |
| | cmdlnend | 在命令行演示过后进行清除操作 |
| | cmdlnwin | 为运行命令行演示而设的演示路径 |
| | finddemo | 查找各个工具箱的可用的演示 |
| | helpfun | 方便显示帮助文本的效用函数 |
| | pltmat | 在图形窗口中显示矩阵 |
| MATLAB/帮助函数 | bucky | Buckminster Fuller 网格球顶图 |
| | membrane | 产生 MathWorks 标识 |
| | peaks | 两个变量的样本函数 |

**表 C-6 MATLAB 中的基本数学函数**

| 功能类别 | 命令函数 | 说明 |
|---|---|---|
| 三角函数 | acos | 反余弦 |
| | acosh | 反双曲余弦 |
| | acot | 反余切 |
| | acoth | 反双曲余切 |
| | acsc | 反余割 |
| | acsch | 反双曲余割 |
| | asec | 反正割 |
| | asech | 反双曲正割 |

（续）

| 功能类别 | 命令函数 | 说　明 |
| --- | --- | --- |
| | asin | 反正弦 |
| | asinh | 反双曲正弦 |
| | atan | 反正切 |
| | atanh | 反双曲正切 |
| | atan2 | 四个象限反正切 |
| | cos | 余弦 |
| | cosh | 双曲余弦 |
| | cot | 余切 |
| | coth | 双曲余切 |
| | csc | 余割 |
| | csch | 双曲余割 |
| | sec | 正割 |
| | sech | 双曲正割 |
| | sin | 正弦 |
| | sinh | 双曲正弦 |
| | tan | 正切 |
| | tanh | 双曲正切 |
| 指数函数 | exp | 指数 |
| | log | 自然对数 |
| | log2 | 以 2 为底的对数和分割浮点数 |
| | log10 | 常用对数 |
| | nextpow2 | 下一个以 2 为底的更高次幂 |
| | pow2 | 以 2 为底的幂和比例浮点数 |
| | sqrt | 平方根 |
| 复数函数 | abs | 绝对值或者复数的模 |
| | angle | 相角 |
| | complex | 由实部和虚部构造复数 |
| | conj | 复共轭 |
| | cplxpair | 在复共轭对中加入有序数 |
| | imag | 复数虚部 |
| | isreal | 判断数组是否为实型数组 |
| | real | 复数实部 |
| | unwrap | 不展开相位角 |
| 取整及余数 | ceil | 朝正无穷大方向取整 |
| | fix | 朝零方向取整 |
| | floor | 朝负无穷大方向取整 |
| | mod | 模（除后取余） |
| | Rem | 除后余数 |
| | Round | 朝最近的整数取整 |
| | Sign | 符号函数 |

表 C-7 MATLAB 中的基本矩阵和矩阵操作

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 基本矩阵 |  | 规则间隔的矢量 |
|  | eye | 单位数组 |
|  | freqspace | 频率响应的频率间隔 |
|  | Linspace | 线性间隔的矢量 |
|  | Logspace | 对数间隔的矢量 |
|  | Meshgrid | 生成 3 维图形所需要的 X 和 Y 数组 |
|  | Ones | 1 数组 |
|  | rand | 均匀分布的随机数 |
|  | randn | 正态分布的随机数 |
|  | repmat | 复制和平铺数组 |
|  | zeros | 0 数组 |
| 基本数组信息 | disp | 显示矩阵或者文本 |
|  | isempty | 判断矩阵是否为空矩阵 |
|  | isequal | 判断数组是否相等 |
|  | islogical | 判断数组是否为逻辑型数组 |
|  | isnumeric | 判断数组是否为数值型数组 |
|  | length | 矢量的长度 |
|  | logical | 将数值量转换为逻辑量 |
|  | ndims | 维数 |
|  | numel | 元素个数 |
|  | size | 矩阵的大小 |
|  | : | 规则间隔的矢量 |
|  | blkdiag | 从输入参数中构造块对角阵 |
|  | diag | 对角阵和矩阵的对角线 |
| 矩阵操作 | end | 代表矩阵的最后下标 |
|  | find | 查找非零元素的下标 |
|  | flipdim | 沿着指定的维翻转矩阵 |
|  | fliplr | 矩阵的左右翻转 |
|  | flipud | 矩阵的上下翻转 |
|  | ind2sub | 从线性下标中获得多重下标 |
|  | reshape | 改变矩阵的大小 |
|  | rot90 | 将矩阵翻转 90° |
|  | sub2ind | 从多重下标中获得线性下标 |
|  | tril | 提取矩阵的下三角部分 |
|  | triu | 提取矩阵的上三角部分 |
| 特定变量以及常量 | ans | 当前的答案 |
|  | eps | 相对浮点精度 |
|  | i | 虚数单位 |
|  | Inf | 无穷大 |
|  | Isfinite | 判断是否为有限量 |
|  | Isinf | 判断是否为无穷大数 |
|  | Isnan | 判断是否为非数值量 |

（续）

| 功能类别 | 命令函数 | 说　明 |
| --- | --- | --- |
| 特定变量以及常量 | J | 虚数单位 |
| | NaN | 非数值量 |
| | pi | 圆周率值 3.1415926535897.... |
| | Realmax | 最大的正浮点数 |
| | realmin | 最小的负浮点数 |
| | why | 简明的答案 |
| 特殊矩阵 | Compan | 友矩阵 |
| 基本矩阵 | gallery | 几个小的测试矩阵 |
| | Hadamard | Hadamard 矩阵 |
| | hankel | Hankel 矩阵 |
| | hilb | Hilbert 矩阵 |
| | invhilb | 逆 Hilbert 矩阵 |
| | magic | 魔方矩阵 |
| | pascal | Pascal 矩阵 |
| | rosser | 经典的对称特征值测试矩阵 |
| | toeplitz | Toeplitz 矩阵 |
| | vander | Vandermonde 矩阵 |
| | wilkinson | Wilkinson 特征值测试矩阵 |

## 表 C-8　MATLAB 中的泛函和常微分方程求解器

| 功能类别 | 命令函数 | 说　明 |
| --- | --- | --- |
| 最优化和求根 | Fminbnd | 求单变量函数的最小值 |
| | Fminsearch | 用 Nelder-Mead 直接搜索法求多变量函数的最小值 |
| | Fzero | 寻找单变量函数的零点 |
| 最优化选项操作 | Optimget | 获取最优化选项“OPTIONS”的结构 |
| | Optimset | 创建或改变最优化选项“OPTIONS”的结构 |
| 数值积分（定积分） | Dblquad | 计算二重积分 |
| | Quad | 低阶法计算数值积分 |
| | Quadl | 高阶法计算数值积分 |
| 绘图 | Ezcontour | 利用等高线绘图器绘制等高线 |
| | Ezcontourf | 利用填充的等高线绘图器绘图 |
| | Ezmesh | 利用三维网格绘图器绘制网格图 |
| | Ezmeshc | 利用网格-等高线绘图器绘图 |
| | Ezplot | 利用函数绘图器绘图 |
| | Ezplot3 | 利用三维参数曲线绘图器绘图 |
| | Ezpolar | 利用极坐标绘图器绘图 |
| | Ezsurf | 利用三维表面阴影绘图器绘图 |
| | Ezsurfc | 利用表面/等高线绘图器绘图 |
| | Fplot | 绘制函数图 |

（续）

| 功能类别 | 命令函数 | 说明 |
|---|---|---|
| 内联对象 | Argnames | 参数名称 |
| | Char | 将内联函数“INLINE”的对象转换成字符型数组 |
| | Formula | 函数公式 |
| | Inline | 构造“INLINE”内联对象 |
| 常微分方程（简称 ODEs）的初始值问题（如果事先不能确定方程的刚性情况，应当首先尝试 ode45 求解器，然后试用 ode15s 求解器） | ode15s | 利用变阶法求解刚性的常微分方程及微分-代数方程 |
| | ode23 | 利用低阶法求解非刚性微分方程 |
| | ode23s | 利用低阶法求解刚性微分方程 |
| | ode23t | 利用梯形规则求解中等刚度的常微分方程及微分-代数方程 |
| | ode23tb | 利用低阶法求解刚性微分方程 |
| | ode45 | 利用中等阶数法求解非刚性微分方程 |
| | ode113 | 利用变阶法求解非刚性微分方程 |
| 常微分方程的边值问题求解器 | bvp4c | 利用配置法求解常微分方程的两点边值问题 |
| 一阶偏微分方程求解器 | pdepe | 求解抛物线-椭圆型偏微分方程的边值问题 |
| 选项操作 | bvpget | 获取边值问题的选项参数 |
| | bvpset | 创建或改变边值问题的选项结构 |
| | odeget | 获取常微分方程的选项参数 |
| | odeset | 创建或改变常微分方程的选项结构 |
| 输入和输出函数 | bvpinit | 提供 BVP4C 的初始估计值 |
| | deval | 估计微分方程问题的解 |
| | odefile | MATLAB 5.0 版常微分方程的文件语法（已被弃用） |
| | odephas2 | 常微分方程输出函数的二维相平面 |
| | odephas3 | 常微分方程输出函数的三维相平面 |
| | odeplot | 常微分方程输出函数的时间序列 |
| | odeprint | 在命令窗口中打印出常微分方程的输出函数 |
| | pdeval | 通过插值计算由 PDEPE 得到的解 |

### 表 C-9　MATLAB 中的通用命令

| 功能类别 | 命令函数 | 说明 |
|---|---|---|
| 管理命令和函数 | addpath | 将路径添加到 MATLAB 要搜索的路径中 |
| | demo | 运行 MATLAB 的演示程序 |
| | doc | 载入超文本说明 |
| | docopt | 显示帮助文件目录的位置 |
| | genpath | 产生路径 |
| | help | 在线帮助文件 |
| | helpbrowser | 显示 MATLAB 在线帮助浏览器 |
| | helpdesk | 显示 MATLAB 帮助浏览器 |

（续）

| 功能类别 | 命令函数 | 说　明 |
|---|---|---|
| 管理命令和函数 | lasterr | 显示最近的出错信息 |
| | lastwarn | 显示最近的警告信息 |
| | license | 使用 MATLAB 的许可证信息 |
| | lookfor | 通过 help 路径条目搜索关键字 |
| | partiapath | 部分路径 |
| | path | 控制 MATLAB 的搜索路径 |
| | pathtool | 打开图形用户界面以及修改路径 |
| | profile | 启动 m 文件调试和优化的代码 |
| | rehash | 更新函数和文件系统的缓冲 |
| | rmpath | 从搜索路径中删除目录 |
| | support | 打开技术支持网页 |
| | type | 列出 m 文件 |
| | what | 显示当前路径内的 m，mat，mbx 文件目录 |
| | which | 定位函数和文件 |
| 管理变量和工作空间 | clear | 从内存中清除变量和函数 |
| | disp | 显示矩阵或者文本 |
| | length | 矢量的长度 |
| | load | 从磁盘中载入工作空间变量 |
| | memory | 内存容量 |
| | mlock | 防止 m 文件被清除 |
| | munlock | 允许清除 m 文件 |
| | openvar | 打开工作空间变量 |
| | pack | 固定工作空间内存 |
| | save | 保存工作空间变量 |
| | saveas | 将工作空间变量另存为 |
| | size | 矩阵的大小 |
| | who | 列出当前变量 |
| | whos | 列出当前变量的详细信息 |
| | workspace | 显示工作空间浏览器 |
| 与文件和操作系统有关的命令 | ! | 执行操作系统命令 |
| | cd | 改变当前工作目录 |
| | delete | 删除文件 |
| | diary | 保存 MATLAB 任务 |
| | dir | 目录列表 |
| | getenv | 获取环境变量值 |
| | unix | 执行 UNIX 操作命令并返回结果 |
| 控制命令窗口 | 控制命令窗口 | |
| | cedit | 设置命令行参数 |
| | clc | 清除命令窗口 |
| | echo | MATLAB 文件中使用的回显命令 |
| | format | 设置输出格式 |
| | home | 光标置左上角 |
| | more | 在命令窗口中控制分页输出 |

（续）

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 启动和退出 | exit | 退出 MATLAB |
| | finish | 终止 m 文件 |
| | matlab | 启动 MATLAB（该命令仅限于 UNIX 操作系统使用） |
| | matlabrc | 主动启动 m 文件 |
| | quit | 终止 MATLAB 的运行 |
| 一般信息 | hostid | MATLAB 主服务程序的识别代号 |
| | info | MATLAB 系统信息及 Mathworks 公司信息 |
| | subscript | 成为 MATLAB 的订购用户 |
| | Ver | MATLAB 的版本信息 |
| | Version | 获取 MATLAB 版本号 |
| | Web | 指向文件和 Web 浏览器 |
| | Whatsnew | 在 MATLAB 说明书中没有包含的新信息 |

**表 C-10　MATLAB 中的 2 维图形函数**

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 基本的 2 维图（X-Y 图形） | loglog | 用全对数坐标绘制图形 |
| | plot | 绘制线性图形 |
| | plotyy | 双 Y 轴绘图 |
| | polar | 极坐标绘图 |
| | semilogx | 半对数坐标图形（X 轴为对数坐标） |
| | semilogy | 半对数坐标图形（Y 轴为对数坐标） |
| 坐标轴的控制 | axis | 控制坐标轴的刻度和形式 |
| | axes | 在任意位置创建坐标轴 |
| | box | 坐标轴绘图对话框 |
| | grid | 绘制网格线 |
| | hold | 保持当前图形 |
| | subplot | 在子窗口位置创建坐标轴 |
| | zoom | 缩放 2 维图 |
| 图形标注 | gtext | 用鼠标放置文本 |
| | legend | 图形注释 |
| | plotedit | 编辑和标注图形的工具 |
| | texlabel | 由字符串产生 TeX 格式的标记 |
| | text | 文本注释 |
| | title | 图形标题 |
| | xlabel | X 轴标记 |
| | ylabel | Y 轴标记 |
| 硬拷贝和打印 | print | 打印图形或者 SIMULINK 系统，或将图形保存为 m 文件 |
| | printopt | 默认的打印机 |
| | orient | 设置纸张走向 |

表 C-11　MATLAB 中的 3 维图形函数

| 功能类别 | 命令函数 | 说　明 |
|---|---|---|
| 基本的 3 维图 | fill3 | 3 维空间内填充多边形 |
| | mesh | 3 维网格图 |
| | plot3 | 绘制 3 维空间里的线和点 |
| | surf | 3 维表面阴影图 |
| 颜色控制 | brighten | 增量和变暗颜色板 |
| | caxis | 伪色彩坐标轴刻度 |
| | colordef | 设置默认颜色 |
| | colormap | 颜色查询表 |
| | graymon | 设置灰度显示器的图形缺省值 |
| | hidden | 去除网格消隐线方式 |
| | shading | 彩色阴影方式 |
| 光照 | diffuse | 漫反射系数 |
| | lighting | 光照模式 |
| | material | 物质反射模式 |
| | specular | 镜面反射率 |
| | surfl | 具有亮度的 3 维表面阴影图 |
| | surfnorm | 表面的法线 |
| 色彩图 | autumn | 红色和黄色渐变的颜色板 |
| | bone | 以蓝色为基调的浓淡映像表 |
| | colorcube | 增强的 3 次颜色映像表 |
| | cool | 深蓝-—品红 2 色浓淡映像表 |
| | copper | 线性青铜色调映像表 |
| | flag | 红，白，蓝，黑交替变换的颜色映像表 |
| | gray | 线性灰度变换表 |
| | hot | 黑-红-黄-白 4 色映像表 |
| | hsv | 色彩-饱和度映像表 |
| | jet | 一种 HSV 的变体 |
| | lines | 线性颜色映像表 |
| | pink | 淡粉红色浓淡映像表 |
| | prism | 光谱颜色板 |
| | spring | 洋红和黄色渐变的颜色板 |
| | summer | 绿色和黄色渐变的颜色板 |
| | vga | Windows16 色颜色板 |
| | white | 纯白色浓淡映像表 |
| | winter | 蓝色和绿色渐变的颜色板 |
| 透明度 | alim | 透明度（alpha）缩放 |
| | alpha | 透明度（alpha）模式 |
| | alphamap | 透明度（alpha）查询表 |
| | axes | 在任意位置创建坐标轴 |
| | axis | 控制坐标轴的刻度和形式 |
| | box | 坐标轴绘图对话框 |
| | daspect | 数据纵横比 |
| | grid | 绘制网格线 |

（续）

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 坐标轴的控制 | Hold | 保持当前图形 |
| | Pbaspect | 绘图对话框的数据纵横比 |
| | Subplot | 在子窗口位置创建坐标轴 |
| | xlim | X 轴的范围 |
| | ylim | Y 轴的范围 |
| | zlim | Z 轴的范围 |
| | zoom | 在 2 维图形上进行缩放操作 |
| 视图控制 | rotate3d | 交互地旋转 3 维视图 |
| | view | 指定 3 维视图的观察点 |
| | viewmtx | 产生视点变换矩阵 |
| 照相机的控制 | campos | 照相机位置 |
| | camproj | 照相机投影 |
| | camtarget | 照相机目标 |
| | camup | 照相机矢量 |
| | camva | 照相机视角 |
| 照相机的高级控制 | camdolly | 移动照相机和目标位置 |
| | cameratoolbar | 交互地操作照相机 |
| | camlookat | 移动照相机和目标以观察指定的对象 |
| | camorbit | 照相机目标轨迹 |
| | campan | 旋转照相机目标 |
| | camroll | 参照坐标轴旋转照相机 |
| | camzoom | 照相机镜头伸缩 |
| 高级光照控制 | camlight | 创建或者获取光的位置 |
| | lightangle | 光的球坐标位置 |
| 图形标注 | colorbar | 显示颜色条（颜色刻度） |
| | gtext | 用鼠标放置文本 |
| | plotedit | 实验图编辑和标注工具 |
| | text | 文本注释 |
| | title | 图形标题 |
| | xlabel | X 轴标记 |
| | ylabel | Y 轴标记 |
| | zlabel | Z 轴标记 |
| 硬拷贝和打印 | orient | 设置纸张走向 |
| | print | 打印图形或者 SIMULINK 系统，或者将图形保存为 m 文件 |
| | printopt | 默认的打印机 |
| | vrml | 将图形保存成 VRML 2.0 文件 |

**表 C-12 MATLAB 中的图形句柄函数**

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 图形窗口的创建和控制 | Clf | 清除当前图形 |
| | Close | 关闭图形 |
| | Figure | 创建图形窗口 |

（续）

| 功能类别 | 命令函数 | 说　明 |
| --- | --- | --- |
| 图形窗口的创建和控制 | Gcf | 获取当前图形的句柄 |
| | Openfig | 打开已保存文件的新备份或者引发已保存文件中已经存在的备份 |
| | Refresh | 刷新图形 |
| | Shg | 显示图形窗口 |
| 轴的创建与控制 | axes | 在任意位置创建坐标轴 |
| | axis | 控制坐标轴的刻度和形式 |
| | box | 坐标轴对话框 |
| | caxis | 控制伪色彩坐标轴的比例 |
| | cla | 清除当前坐标轴 |
| | gca | 获取当前轴的句柄 |
| | hold | 保持当前图形 |
| | ishold | 返回图形保持状态 |
| | subplot | 在子窗口位置创建坐标轴 |
| 处理图形对象 | axes | 创建坐标轴 |
| | figure | 创建图形窗口 |
| | image | 创建图像 |
| | light | 创建光照 |
| | line | 创建线 |
| | patch | 创建图形填充块 |
| | rectangle | 创建矩形，倒角的矩形或者椭圆 |
| | Surface | 创建面 |
| | text | 创建文本 |
| | uicontextmenu | 创建用户关联菜单 |
| | uicontrol | 创建用户界面控件 |
| | uimenu | 创建用户界面菜单 |
| 处理图形选项 | copyobj | 检查是否是应用程序定义的数据 |
| | delete | 删除对象 |
| | drawnow | 刷新正在挂起的图形事件 |
| | findobj | 查找指定属性值的对象 |
| | gcbf | 获取当前调回图形的句柄 |
| | gcbo | 获取当前调回对象的句柄 |
| | gco | 获取当前对象的句柄 |
| | get | 获取对象属性 |
| | getappdata | 获取应用程序定义的数据值 |
| | isappdata | 检查应用程序定义的数据是否存在 |
| | rmappdata | 删除应用程序定义的数据 |
| | reset | 重新设置对象属性 |
| | set | 设置对象属性 |
| | setappdata | 指定应用程序定义的数据 |

（续）

| 功能类别 | 命令函数 | 说　明 |
| --- | --- | --- |
| 硬拷贝和打印 | orient | 设置纸张走向 |
| | print | 打印图形或者 SIMULINK 系统，或者将图形保存为 m 文件 |
| | printopt | 默认的打印机 |
| 其他函数 | closereq | 图形关闭请求函数 |
| | ishandle | 检查是否为图形句柄 |
| | newplot | NextPlot 属性的 m 文件序文 |
| ActiveX 客户程序函数（仅限于 PC 机） | actxcontrol | 创建一个 ActiveX 控件 |
| | actxserver | 创建一个 ActiveX 服务器 |

**表 C-13　MATLAB 中的文件输入/输出函数**

| 功能类别 | 命令函数 | 说　明 |
| --- | --- | --- |
| 文件输入/输出函数 | dlmread | 从以 ASCII 码限界的文件中读一矩阵 |
| | dlmwrite | 按 ASCII 码限界的文件格式写一矩阵 |
| | importdata | 加载工作空间变量磁盘文件 |
| | load | 从 MATLAB（MAT）文件加载工作空间 |
| | wk1read | 读 WK1 文件 |
| | wk1write | 写 WK1 文件 |
| | xlsread | 读 XLS 文件 |
| 图像文件输入/输出 | imfinfo | 返回图形文件的信息 |
| | imread | 读取图形文件 |
| | imwrite | 写入图形文件 |
| 音频文件输入/输出 | auread | 读 NeXT/SUN （AU）音频文件 |
| | auwrite | 写 NeXT/SUN （AU）音频文件 |
| | wavread | 读 Microsoft WAVE （WAV）音频文件 |
| | wavwrite | 写 Microsoft WAVE （WAV）音频文件 |
| 视频文件输入/输出 | Avifile | 创建一个新的 AVI 文件 |
| | Aviinfo | 返回 AVI 文件的信息 |
| | Aviread | 读动画（AVI）文件 |
| | movie2avi | 由 MATLAB 动画创建 AVI 动画 |
| 格式文件输入/输出 | fgetl | 从文件中读取行并放弃换行符 |
| | fgets | 从文件中读取行并保持换行符 |
| | fprintf | 将格式化数据写入文件 |
| | fscanf | 从文件中读取格式化数据 |
| | input | 提示用户输入 |
| | textread | 从文本文件中读取格式化数据 |
| 字符串变换 | sprintf | 将格式化数据写入到字符串 |
| | sscanf | 从格式化字符串中读取数据 |
| | strread | 从字符串中读取格式化数据 |

（续）

| 功能类别 | 命令函数 | 说　明 |
|---|---|---|
| 文件的打开与关闭 | 文件的打开与关闭 | |
| | fclose | 关闭文件 |
| | fopen | 打开文件 |
| 二进制文件的输入/输出 | fread | 从文件中读取二进制数据 |
| | fwrite | 将二进制数据写入文件 |
| 文件的定位 | feof | 测试文件尾 |
| | ferror | 查询文件出错的状态 |
| | frewind | 将一个打开文件的指针反绕 |
| | Fseek | 设置文件的位置指针 |
| | ftell | 获取文件的位置指针 |
| 文件名的操作 | fileparts | 文件名分解 |
| | filesep | 对本操作平台进行目录隔离 |
| | fullfile | 由文件的各组成部分构建全文件名 |
| | matlabroot | 安装 MATLAB 的根目录 |
| | mexext | 本操作平台下的 MEX 文件扩展名 |
| | partialpath | 部分路径名 |
| | pathsep | 对本操作平台进行路径隔离 |
| | Prefdir | 优先目录名 |
| | Tempdir | 获取临时目录 |
| | tempname | 获取临时文件 |
| | hdf | 由 MEX 文件界面转换到 HDF 库 |
| | hdfan | 由 MATLAB Gateway 到多文件注释界面 |
| | hdfdf24 | 由 MATLAB Gateway 到 HDF 光栅图像界面 |
| | hdfdfr8 | 由 MATLAB Gateway 到 HDF8 位光栅图像界面 |
| | hdfh | 由 MATLAB Gateway 到 HDF H 界面 |
| | hdfhd | 由 MATLAB Gateway 到 HDF HD 界面 |
| | hdfhe | 由 MATLAB Gateway 到 HDF HE 界面 |
| | hdfml | MATLAB-HDF 路径设备 |
| | hdfsd | 由 MATLAB Gateway 到多文件科学数据集界面 |
| | hdfv | 由 MATLAB Gateway 到 HDF V（Vgroup）界面 |
| | hdfvf | 由 MATLAB Gateway 到 HDF VF（Vdata）界面 |
| | hdfvh | 由 MATLAB Gateway 到 HDF VH（Vdata）界面 |
| | hdfvs | 由 MATLAB Gateway 到 HDF VS（Vdata）界面 |
| HDF-EOS 库界面帮助 | hdfgd | 由 MATLAB Gateway 到 HDF-EOS 网格界面 |
| | hdfpt | 由 MATLAB Gateway 到 HDF-EOS 点界面 |
| | hdfsw | 由 MATLAB Gateway 到 HDF-EOS 扫描到 interface |
| 串行端口支持 | serial | 构建串行端口对象 |

（续）

| 功能类别 | 命令函数 | 说明 |
|---|---|---|
| 命令窗口输入/输出函数 | clc | 清除命令窗口中的内容 |
| | disp | 显示数组 |
| | home | 将光标置于初始位置 |
| | input | 提示用户进行屏幕输入 |
| | pause | 暂停（等待用户响应） |
| 图形文件的绘图编辑及输出帧支持函数 | hgload | 从文件加载一个图形句柄对象 |
| | hgsave | 向文件保存一个 HG（图形句柄）对象继承 |
| 实用程序 | str2rng | 将电子表格范围字符串转换为数值型数组 |
| | wk1const | WK1 记录类型定义 |
| | wk1wrec | 写 WK1 记录标题 |

**表 C-14　MATLAB 中的语言结构**

| 功能类别 | 命令函数 | 说明 |
|---|---|---|
| 流程控制 | break | 终止执行 for 循环或 while 循环 |
| | case | 条件转换 |
| | catch | 开始捕获块 |
| | continue | 将流程传到下一个 for 循环或者 while 循环 |
| | else | 有条件地执行语句 |
| | elseif | 有条件地执行语句 |
| | end | 表明语句执行结束 |
| | for | 指定循环次数的循环语句 |
| | if | 条件执行语句 |
| | otherwise | 默认的 switch 语句条件 |
| | return | 返回到调用函数 |
| | switch | 基于表达式的多种情况 |
| | try | 开始检测块 |
| | while | 不确定循环次数的循环语句 |
| 赋值和执行 | assignin | 在工作空间中分配变量 |
| | builtin | 从加载的方法中执行 built-in 函数 |
| | eval | 执行 MATLAB 表达式所描述的字符串 |
| | evalc | 求 MATLAB 表达式所描述的字符串的值 |
| | evalin | 求工作空间中表达式的值 |
| | feval | 求函数值 |
| | run | 运行命令文件 |
| 命令，函数以及变量 | exist | 检查变量或者函数是否已经被定义 |
| | function | 添加新的函数 |
| | global | 定义全局变量 |
| | isglobal | 检查变量是否为全局变量 |
| | iskeyword | 检查输入是否是一个关键字 |

（续）

| 功能类别 | 命令函数 | 说　明 |
|---|---|---|
| 命令，函数以及变量 | isvarname | 检查一个有效的变量名 |
| | lists | 由逗号作为分隔符的列表 |
| | mfilename | 当前所执行的 m 文件的名称 |
| | mislocked | 检查 m 文件是否不能被清除 |
| | mlock | 防止清除 m 文件 |
| | munlock | 允许清除 m 文件 |
| | persistent | 定义固定变量 |
| | precedence | MATLAB 的优先级顺序 |
| | script | 和 MATLAB 有关的命令和 m 文件 |
| 参数处理 | inputname | 输入参数的名称 |
| | nargchk | 确定输入参数的个数 |
| | nargin | 函数的输入参数的个数 |
| | nargout | 函数的输出参数的个数 |
| | nargoutchk | 确定输出参数的个数 |
| | varargin | 输入参数列表的变量长度 |
| | varargout | 输出参数列表的变量长度 |
| 信息显示 | disp | 显示数组 |
| | display | 重载函数以显示一个数组 |
| | error | 中断函数并显示出错信息 |
| | fprintf | 显示数据格式的信息 |
| | lasterr | 最近的出错信息 |
| | lastwarn | 最近的警告信息 |
| | sprintf | 将有格式数据写给一个字符串 |
| | warning | 显示警告信息 |
| 交互输入 | input | 提示用户输入 |
| | keyboard | 在 m 文件中调用键盘 |
| | pause | 暂时停止执行程序 |
| | uicontrol | 创建用户交互控制 |
| | uimenu | 创建用户交互菜单 |

**表 C-15　MATLAB 中的矩阵函数以及数值线性代数**

| 功能类别 | 命令函数 | 说　明 |
|---|---|---|
| 矩阵分析 | Det | 行列式 |
| | Norm | 矩阵或者矢量的范数 |
| | Normest | 矩阵的 2 范数估计 |
| | Null | 零空间 |
| | orth | 正交化 |
| | rank | 矩阵的秩 |
| | rref | 缩减行阶梯格式 |

（续）

| 功 能 类 别 | 命 令 函 数 | 说 明 |
| --- | --- | --- |
| 矩阵分析 | subspace | 两个子空间之间的夹角 |
| | trace | 对角线元素的和（即矩阵的迹） |
| 线性方程 | \ | 线性方程求解 |
| | / | 线性方程求解 |
| | chol | 矩阵的 Cholesky 分解 |
| | cholinc | 矩阵的非完全 Cholesky 分解 |
| | cond | 计算矩阵的条件数 |
| | condest | 矩阵的 1 范数条件数估计 |
| | inv | 求矩阵的逆 |
| | lscov | 协方差已知条件下的最小二乘系数 |
| | lsqnonneg | 非负最小平方 |
| | lu | 矩阵的 LU 分解 |
| | luinc | 矩阵的非完全 LU 分解 |
| | normest1 | 矩阵的 1 范数估计 |
| | pinv | 矩阵的伪逆 |
| | qr | 矩阵的正交-三角分解（即 QR 分解） |
| | rcond | LAPACK 倒数条件估计量 |
| 特征值和奇异值 | condeig | 关于特征值的条件数 |
| | eig | 求矩阵的特征值和特征矢量 |
| | eigs | 求矩阵的某些特征值 |
| | gsvd | 矩阵的广义奇异值分解 |
| | hess | 矩阵的 Hessenberg 形式 |
| | poly | 求特征多项式 |
| | polyeig | 多项式特征值问题 |
| | qz | 广义特征值的 QZ 分解 |
| | schur | 矩阵的 Schur 分解 |
| | svd | 矩阵的奇异值分解 |
| | svds | 求矩阵的某些奇异值 |
| 矩阵函数 | balance | 改变对角线的比例以提高特征值的精度（即对矩阵进行平衡处理） |
| | cdf2rdf | 变复分块对角阵为实分块对角形式 |
| | cholupdate | 由秩为 1 更正为 Cholesky 分解 |
| | expm | 矩阵的指数 |
| | funm | 一般矩阵的计算 |
| | logm | 矩阵的对数 |
| | planerot | 矩阵的 Givens 平面旋转 |
| | qrdelete | 从 QR 分解中删除列 |
| | qrinsert | 在 QR 分解中插入列 |
| | qrupdate | 由秩为 1 更正为 QR 分解 |
| | rsf2csf | 变实分块对角阵为复分块对角形式 |
| | sqrtm | 矩阵开平方根 |

表 C-16　MATLAB 中的操作符和特殊字符

| 功能类别 | 命令函数 | 说明 |
|---|---|---|
| 数学操作符 | + | 加 |
| | - | 减 |
| | * | 矩阵乘法 |
| | .* | 数组乘法 |
| | ^ | 矩阵幂 |
| | .^ | 数组幂 |
| | / | 矩阵右除或者斜杠 |
| | \ | 矩阵左除或者反斜杠 |
| | ./ | 数组右除 |
| | .\ | 数组左除 |
| | kron | Kronecker 张量积 |
| 关系操作符 | == | 相等 |
| | ~= | 不等 |
| | < | 小于 |
| | > | 大于 |
| | <= | 小于等于 |
| | >= | 大于等于 |
| 逻辑操作符 | & | 逻辑与 |
| | \| | 逻辑或 |
| | ~ | 逻辑非 |
| | all | 矢量的所有元素为真，则其值为真 |
| | any | 矢量的任一元素为真，则其值为真 |
| | xor | 逻辑异或 |

表 C-17　MATLAB 中的插值和多项式

| 功能类别 | 命令函数 | 说明 |
|---|---|---|
| 数据插值 | griddata | 数据网格和表面拟合 |
| | griddata3 | 三维数据网格和超平面拟合 |
| | griddatan | 数据网格和超平面拟合（维数≥2） |
| | interpft | 用 FFT 法进行一维插值 |
| | interp1 | 一维插值（一维查表） |
| | interp1q | 快速一维线性插值 |
| | interp2 | 二维插值（二维查表） |
| | interp3 | 三维插值（三维查表） |
| | interpn | N 维插值（N 维查表） |
| | pchip | 分段立方 Hermite 插值多项式 |
| 样条插值 | ppval | 分段多项式计算 |
| | spline | 3 次样条插值 |

（续）

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 几何分析 | convhull | 凸包 |
| | convhulln | N 维凸包 |
| | delaunay | Delaunay 三角测量 |
| | delaunay3 | 三维 Delaunay 布置 |
| | delaunayn | Delaunay 布置 |
| | dsearch | 搜索最近点的 N 维 Delaunay 布置 |
| | inpolygon | 判断点是否位于多边形区域内 |
| | polyarea | 多边形面积 |
| | rectint | 矩形的交叉面积 |
| | tsearch | 最近点三角搜索 |
| | tsearchn | （n 维）最近点三角搜索 |
| | voronoi | Voronoi 图表 |
| | voronoin | N 维 Voronoi 图表 |
| 多项式 | conv | 多项式乘法 |
| | deconv | 多项式除法 |
| | poly | 构造具有指定根的多项式 |
| | polyder | 微分多项式 |
| | polyfit | 数据的多项式拟合 |
| | polyint | 积分解析多项式 |
| | polyval | 多项式计算 |
| | polyvalm | 带矩阵变量的多项式计算 |
| | residue | 部分分式展开（即留数计算） |
| | roots | 求多项式的根 |

**表 C-18　MATLAB 中的稀疏矩阵函数**

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 基本稀疏矩阵 | spdiags | 由对角线组成的稀疏矩阵 |
| | speye | 稀疏单位阵 |
| | sprand | 均匀分布的稀疏随机矩阵 |
| | sprandn | 正态分布的稀疏随机矩阵 |
| | sprandsym | 对称的随机稀疏矩阵 |
| 满阵转和稀疏矩阵间的转换 | find | 找出非零元素的标号 |
| | full | 将稀疏矩阵转换为满阵 |
| | sparse | 创建稀疏矩阵 |
| | spconvert | 输入外部格式的（即非 MATLAB 格式的）稀疏矩阵 |
| 稀疏矩阵的使用 | issparse | 判断矩阵是否为稀疏矩阵 |
| | nnz | 非零元素的个数 |
| | nonzeros | 非零的矩阵元素 |

（续）

| 功能类别 | 命令函数 | 说明 |
|---|---|---|
| 稀疏矩阵的使用 | nzmax | 分配给非零元素的存储量 |
| | spalloc | 为稀疏矩阵分配空间 |
| | spfun | 函数只对非零元素起作用 |
| | spones | 用“1”代替稀疏矩阵中的非零元素 |
| | spy | 显示稀疏矩阵 |
| 排序算法 | colamd | 稀疏矩阵列的近似最小度排列 |
| | colmmd | 稀疏矩阵列的最小度排列 |
| | colperm | 按照列进行排列 |
| | dmperm | 按照 Dulmage-Mendelsohn 法进行排列 |
| | randperm | 随机排列矢量 |
| | symamd | 稀疏矩阵列的对称近似最小度排列 |
| | symmmd | 稀疏矩阵列的对称最小度排列 |
| | symrcm | 稀疏矩阵列的对称逆 Cuthill-McKee 序排列 |
| 线性代数 | cholinc | 不完全 Cholesky 分解 |
| | condest | 1 范数的条件数估计 |
| | eigs | 利用 ARPACK 软件包求解少量特征值 |
| | luinc | 不完全 LU 分解 |
| | normest | 2 范数估计 |
| | sprank | 结构化秩 |
| | svds | 利用 eigs 命令求解少量奇异值 |
| | bicg | 双共轭梯度法 |
| | bicgstab | 稳定的双共轭梯度法 |
| | cgs | 平方共轭梯度法 |
| | gmres | 广义的最小残差法 |
| 线性方程（迭代法） | minres | 最小残差法 |
| | pcg | 预处理共轭梯度法 |
| | qmr | 准最小残差法 |
| | symmlq | 对称 LQ 法 |
| 曲线图（树型）操作 | etree | 求矩阵的消元树 |
| | etreeplot | 画消元树图 |
| | gplot | 按照图论的方法绘制图形 |
| | treelayout | 显示一个或者多个结构树 |
| | treeplot | 画结构树 |
| 其他 | spaugment | 形成最小二乘增广系统 |
| | spparms | 为稀疏矩阵处理过程设置参数 |
| | symbfact | 符号分解分析 |

表 C-19 MATLAB 中的特殊数学函数

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 特殊数学函数 | airy | Airy 函数 |
| | besselh | 第三类 Bessel 函数（Hankel 函数） |
| | besseli | 改进的第一类 Bessel 函数 |
| | besselj | 第一类 Bessel 函数 |
| | besselk | 改进的第二类 Bessel 函数 |
| | bessely | 第二类 Bessel 函数 |
| | beta | β 函数 |
| | betainc | 非完全的 β 函数 |
| | betaln | β 函数的对数 |
| | cross | 矢量的叉积 |
| | dot | 矢量的点积 |
| | ellipj | Jacobi（雅可比）椭圆函数 |
| | ellipke | 完全椭圆积分 |
| | erf | 误差函数 |
| | erfc | 互补误差函数 |
| | erfcx | 比例互补误差函数 |
| | erfinv | 逆误差函数 |
| | expint | 指数积分函数 |
| | gamma | γ 函数 |
| | gammainc | 非完全的 γ 函数 |
| | gammaln | γ 函数的对数 |
| | legendre | 连带的 Legendre 函数 |
| 数论函数 | factor | 素数因子（质因数分解） |
| | factorial | 阶乘函数 |
| | gcd | 最大公约数 |
| | isprime | 判断是否为素数 |
| | lcm | 最小公倍数 |
| | nchoosek | 从 N 个元素中取 K 个元素的所有组合 |
| | perms | 所有的可能排列 |
| | primes | 产生素数序列 |
| | rat | 有理逼近 |
| | rats | 有理输出 |
| 坐标变换 | cart2pol | 笛卡儿坐标转换成极坐标 |
| | cart2sph | 笛卡儿坐标转换成球坐标 |
| | hsv2rgb | 把色彩饱和值颜色转换为 RGB 颜色 |
| | pol2cart | 极坐标转换成笛卡儿坐标 |
| | rgb2hsv | 把 RGB 颜色转换为色彩饱和值颜色 |
| | sph2cart | 球坐标转换成笛卡儿坐标 |

表 C-20　MATLAB 中的特殊图形函数

| 功能类别 | 命令函数 | 说明 |
|---|---|---|
| 特殊的二维图形 | area | 面积填充图 |
| | bar | 条形图 |
| | barh | 水平条形图 |
| | comet | 彗星图 |
| | compass | 画区域图 |
| | errorbar | 误差条形图 |
| | ezplot | 利用函数绘图器绘图 |
| | ezpolar | 利用三维参数曲线绘图器绘图 |
| | feather | 绘制箭头图 |
| | fill | 2 维填充的多边形 |
| | fplot | 绘制函数图形 |
| | hist | 直方图 |
| | pareto | 绘制排列图 |
| | pie | 绘制饼状图 |
| | plotmatrix | 绘制矩阵的分散图 |
| | rose | 角度直方图 |
| | scatter | 分散图 |
| | stairs | 阶梯图 |
| | stem | 离散序列图或者杆状图 |
| 2 维和 3 维等高线绘图 | clabel | 在等高线图上增加高度标记 |
| | contour | 绘制等高线图 |
| | contourf | 填充的等高线图 |
| | contour3 | 绘制 3 维等高线图 |
| | ezcontour | 利用等高线绘图器绘制等高线图形 |
| | ezcontourf | 利用等高线绘图器绘制填充的等高线图形 |
| | pcolor | 画伪色彩图形 |
| | voronoi | 画 Voronoi 图 |
| 特殊的 3 维图形 | bar3 | 3 维条形图 |
| | bar3h | 水平方向的 3 维条形图 |
| | comet3 | 3 维彗星图 |
| | ezgraph3 | 绘制满足一般要求的表面图 |
| | ezmesh | 利用 3 维网格绘图器绘图 |
| | ezmeshc | 利用网格/等高线绘图器绘图 |
| | ezplot3 | 利用 3 维参数曲线绘图器绘图 |
| | ezsurf | 利用 3 维表面阴影绘图器绘图 |
| | ezsurfc | 利用 3 维表面/等高线绘图器绘图 |
| | meshc | 网格/等高线混合图形 |
| | meshz | 带参考平面的 3 维网格图 |

（续）

| 功能类别 | 命令函数 | 说明 |
|---|---|---|
| 特殊的3维图形 | pie3 | 3维饼状图 |
| | ribbon | 带状图 |
| | scatter3 | 3维离散图 |
| | stem3 | 3维杆图 |
| | surfc | 表面/等高线混合图形 |
| | trimesh | 三角网格图 |
| | trisurf | 三角曲面图 |
| | waterfall | 落差图 |
| 体积和矢量可视化 | coneplot | 3 维锥形图 |
| | contourslice | 在截面上绘制等高线 |
| | curl | 计算矢量场的旋度后角速度 |
| | divergence | 计算矢量场的散度 |
| | interpstreamspeed | 计算流场的顶点 |
| | isocaps | 计算等值面的末端 |
| | isocolors | 计算等值面的颜色 |
| | isonormals | 计算等值面顶点的范数 |
| | isosurface | 从体积数据中提取等值面数据 |
| | quiver | 2维箭头图 |
| | quiver3 | 3维箭头图 |
| | reducepatch | 减少填补面的数值 |
| | reducevolume | 减少体积数据集的数值 |
| | shrinkfaces | 减少填补面的尺寸 |
| | Slice | 立体截面图 |
| | smooth3 | 光滑3维数据 |
| | Streamline | 绘制2维或3维矢量的流线数据 |
| | stream2 | 计算2维流线数据 |
| | stream3 | 计算3维流线数据 |
| | streamparticles | 画出流极小量 |
| | streamribbon | 绘制流带图 |
| | streamslice | 在截平面上绘制流图 |
| | streamtube | 绘制流的管状图 |
| | subvolume | 提取体积数据集合的子集 |
| | vissuite | 可视序列 |
| | volumebounds | 返回体积数据的坐标和颜色范围 |
| 图形显示及文件输入/输出 | brighten | 增量和变暗颜色板 |
| | colorbar | 显示颜色条（代表颜色比例） |
| | colormap | 颜色查找表 |
| | contrast | 调节对比度颜色板以增强图像的对比效 |
| | gray | 线性灰度变换表 |

（续）

| 功能类别 | 命令函数 | 说　明 |
|---|---|---|
| 图形显示及文件输入/输出 | image | 显示图像 |
| | imagesc | 缩放数据并显示成图像 |
| | imfinfo | 图形文件的有关信息 |
| | imread | 从图形文件读入图像 |
| | imwrite | 将图像写入图形文件 |
| 动画 | capture | 抓取当前屏幕的图形 |
| | frame2im | 将动画帧转换为编入索引的图像 |
| | getframe | 获取动画帧 |
| | im2frame | 将编入索引的图像转换为动画帧格式 |
| | movie | 播放所记录的动画帧 |
| | moviein | 动画帧内存初始化 |
| | rotate | 沿着指定的原点和方向旋转对象 |
| 与颜色有关的函数 | colstyle | 从字符串中分出颜色和样式 |
| | ind2rgb | 将编入索引的图像转换为 RGB 图像 |
| | rgbplot | 绘制颜色板图形 |
| | spinmap | 旋转颜色板 |
| 立体建模 | cylinder | 产生圆柱体 |
| | ellipsoid | 产生椭圆体 |
| | patch | 创建图形填充块 |
| | sphere | 产生球 |
| | surf2patch | 将表面数据转换为图形填充块数据 |

表 C-21　MATLAB 中的字符串函数

| 功能类别 | 命令函数 | 说　明 |
|---|---|---|
| 一般函数 | blanks | 空串 |
| | cellstr | 由特征数组创建字符串单元数组 |
| | char | 创建字符数组（字符串） |
| | deblank | 删除尾部的空串 |
| | double | 将字符串转换为数值型编码 |
| | eval | 执行由 MATLAB 表达式组成的字符串 |
| 字符串检查 | iscellstr | 检查数组是否为字符串型单元数组 |
| | ischar | 检查数组（字符串）是否为字符型 |
| | isletter | 判断变量是否为字母 |
| | isspace | 检查变量是否为空字符 |
| | findstr | 在字符串中查找另外一个子串 |
| | lower | 把字符串变为小写 |
| | strcat | 连接字符串 |
| | strcmp | 比较字符串 |
| | strcmpi | 忽略大小写，比较字符串 |
| | strjust | 验证字符型数组 |

（续）

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 字符串检查 | strmatch | 查找符合要求的字符串 |
| | strncmp | 比较字符串的前N个字符 |
| | strncmpi | 忽略大小写，比较字符串的前N个字符 |
| | strrep | 用另外一个字符串代替当前的字符串 |
| | strtok | 在字符串中查找标记 |
| | strvcat | 竖直连接字符串 |
| | upper | 把字符串变为大写 |
| 字符串与数值之间的转换 | int2str | 变整数为字符串 |
| | mat2str | 变矩阵为可计算的字符串 |
| | num2str | 变数值为字符串 |
| | sprintf | 变数值为格式控制下的字符串 |
| | sscanf | 变字符串为格式控制下的数值 |
| | str2double | 变字符串为双精度值 |
| | str2num | 变字符串矩阵为数值型数组 |
| 基本进制转换 | base2dec | 变以B为基的字符串为十进制整数 |
| | bin2dec | 变二进制字符串为十进制整数 |
| | dec2base | 将十进制整数变成以B为基的字符串 |
| | dec2bin | 变十进制整数为二进制字符串 |
| | dec2hex | 变十进制整数为十六进制字符串 |
| | hex2dec | 变十六进制字符串为十进制整数 |
| | hex2num | 变十六进制数为IEEE标准下的浮点数 |

**表 C-22　MATLAB 中的时间和日期函数**

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 当前日期和时间 | c lock | 以日期矢量形式表示的当前日期和时间 |
| | date | 以日期字符串形式表示的当前日期 |
| | now | 以日期数形式表示的当前日期和时间 |
| 基本函数 | datenum | 日期的数字序列格式 |
| | datestr | 日期的字符串格式 |
| | datevec | 日期的矢量格式 |
| 日期函数 | calendar | 日历 |
| | datetick | 有格式的标注日期 |
| | eomday | 月末的最后一天 |
| | weekday | 星期 |
| 计时函数 | cputime | 以秒表示的CPU时间 |
| | etime | 时间差 |
| | pause | 按秒等待 |
| | tic | 开始执行秒表计时 |
| | toc | 结束秒表计时 |

表 C-23　MATLAB 中的图形用户界面（GUI）工具

| 功能类别 | 命令函数 | 说　明 |
|---|---|---|
| GUI 函数 | dragrect | 用鼠标拖动“异或”矩形 |
| | selectmoveresize | 交互地选择，移动，改变大小或者拷贝对象 |
| | uicontrol | 创建用户界面控件 |
| | uimenu | 创建用户界面菜单 |
| | uirestore | 恢复图形的交互状态 |
| | uiresume | 继续执行先前停止执行的程序 |
| | uistack | 控制对象的堆栈顺序 |
| | uisuspend | 挂起图形的交互状态 |
| | uiwait | 停止执行程序并等待恢复 |
| | waitforbuttonpress | 在图形中等待按键/按钮 |
| | waitfor | 停止执行程序并等待事件 |
| GUI 设计工具 | align | 排列 UI 控件和轴 |
| | guide | 设计 GUI |
| | inspect | 检查对象的属性 |
| | propedit | 编辑属性 |
| 对话框 | axlimdlg | 轴范围对话框 |
| | dialog | 创建对话框 |
| | errordlg | 出错对话框 |
| | helpdlg | 帮助对话框 |
| | imageview | 利用缩放图形显示图像 |
| | inputdlg | 输入对话框 |
| | listdlg | 列表选择对话框 |
| | menu | 为用户输入产生选择菜单 |
| | movieview | 利用重放按钮显示动画图形 |
| | msgbox | 消息对话框 |
| | pagedlg | 页面位置对话框 |
| | pagesetupdlg | 页面设置对话框 |
| | printdlg | 打印对话框 |
| | printpreview | 预览待打印的图形 |
| | questdlg | 提问对话框 |
| | soundview | 以图形和播放的形式显示声音 |
| | uigetfile | 标准的打开文件对话框 |
| | uigetpref | 支持优先级的提问对话框 |
| | uiimport | 为输入数据（Import Wizard）而启动 GUI |
| | uiload | 显示打开文件对话框并在结果中调用“load” |
| | uiopen | 显示打开文件对话框并在结果中调用“open” |
| | uiputfile | 标准的保存文件对话框 |
| | uisave | 显示打开文件对话框并在结果中调用“save” |
| | uisetcolor | 颜色选择对话框 |
| | uisetfont | 字体选择对话框 |

（续）

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 对话框 | waitbar | 显示等待条 |
| | warndlg | 警告对话框 |
| 菜单函数 | makemenu | 创建菜单结构 |
| | menubar | 对 MenuBar 属性而言计算机所依赖的默认设置 |
| | umtoggle | 切换 uimenu 对象的“checked”状态 |
| | winmenu | 给“窗口”菜单项目创建子菜单 |
| 工具栏按钮组件函数 | btndown | 在工具栏按钮组件中压下按钮 |
| | btnresize | 改变按钮组件的大小 |
| | btngroup | 创建工具栏按钮组件 |
| | btnpress | 工具栏按钮组件的按钮管理 |
| | btnstate | 提问工具栏按钮组件的状态 |
| | btnup | 在工具栏按钮组件中弹起按钮 |
| 优先权 | addpref | 添加优先权 |
| | getpref | 获取优先权 |
| | ispref | 测验优先权的存在情况 |
| | rmpref | 去除优先权 |
| | setpref | 设置优先权 |
| 其他函数 | allchild | 获取所有对象的子类 |
| | clipboard | 输给系统剪贴板以及来自系统剪贴板的拷贝和粘贴字符串 |
| | edtext | 轴文本对象的交互编辑 |
| | findall | 搜索所有的对象 |
| | findfigs | 搜索图形超出屏幕的部分 |
| | getptr | 获取图形指针 |
| | getstatus | 获取图形中文本字符串的状态 |
| | guidata | 储存或恢复应用数据 |
| | guihandles | 返回句柄的一个结构 |
| | hidegui | 隐藏/显示 GUI |
| | listfonts | 从单元数组中获取可用的系统字体列表 |
| | movegui | 将 GUI 移到屏幕的指定部分 |
| | overobj | 获取指针结束时的对象句柄 |
| | popupstr | 获取上拉式菜单的选项字符串 |
| | remapfig | 变换图形对象的位置 |
| | setptr | 设置图形指针 |
| | setstatus | 设置图形中文本字符串的状态 |
| | uiclearmode | 清除当前激活的交互模式 |

**表 C-24　MATLAB 中的版本控制函数**

| 功能类别 | 命令函数 | 说明 |
| --- | --- | --- |
| 版本控制命令 | checkin | 将文件登入版本控制系统 |
| | checkout | 在版本控制系统中检查文件 |
| | undocheckout | 撤消在版本控制系统中检查文件 |

（续）

| 功能类别 | 命令函数 | 说　明 |
| --- | --- | --- |
| 特殊的版本控制 | clearcase | 用 ClearCase 实现版本控制操作 |
| | customverctrl | 定制版本控制模板 |
| | pvcs | 用 PVCS 实现版本控制操作 |
| | rcs | 用 RCS 实现版本控制操作 |
| | sourcesafe | 用 Visual SourceSafe 实现版本控制操作 |

表 C-25　MATLAB 中的 Windows 操作系统界面文件（DDE/ActiveX）

| 功能类别 | 命令函数 | 说　明 |
| --- | --- | --- |
| ActiveX 客户程序函数 | actxcontrol | 创建一个 ActiveX 控件 |
| | actxserver | 创建一个 ActiveX 服务器 |
| | winfun\activex | ActiveX 类 |
| 演示 ActiveX | mwsamp | ActiveX 控件创建演示程序 |
| | sampev | ActiveX 服务器的事件处理演示程序 |
| DDE 客户程序函数 | ddeadv | 创建咨询链接 |
| | ddeexec | 发送执行字符串 |
| | ddeinit | 初始化 DDE 对话 |
| | ddepoke | 发送数据到应用程序 |
| | ddereq | 从应用程序请求数据 |
| | ddeterm | 终止 DDE 对话 |
| | ddeunadv | 释放咨询链接 |